PROJECTE FINAL DE CARRERA

ANÀLISIS I IMPLEMENTACIÓ D'UN SINTETIZADOR VIRTUAL AMB PURE DATA

Trets fonamentals de la síntesis substractiva

(ANALYSIS AND IMPLEMENTATION OF SUBTRACTIVE VIRTUAL SYNTHESIZER IN PURE DATA

Main features of subtractive synthesis)

Estudis: Enginyeria Electrònica

Autor: Ferran Galí Romea

Director/a: Giuseppe Torre

Any:2010/2011

I'm here writing only because of my parents.
If they had not been sincerely and strongly supporting me
this dedication could not exist.
I wish to exclusively dedicate to them
these first lines of my music-related final project
Thanks for your love, effort and patience Mum and Dad.

UNIVERSITY *of* LIMERICK

OLLSCOIL LUIMNIGH

Digital Media & Arts Research Centre

Department of Computer Science & Information Systems
Faculty of Science and Engineering

Abstract

FulSynth is a Pure Data self-developed virtual subtractive synthesizer. It has led a researching work concerning synthesizers using a subtractive synthesis method. The theoretical fundamentals of subtractive synthesis are put forward. The self-developed application is compared with three commercial virtual synthesizers and conclusions arise through researching, developing and comparing work.

To understand many of the devices or electronic instruments that have been built until today it is essential to take in consideration subtractive synthesis. Most of the analogue and digital commercial synthesizers use this technique as a resource to produce and manipulate the sound.

The whole work has required previous Pure Data language training and eventually provides a practical, creative and technical understanding of subtractive synthesis sound designing, music manipulating methods.

Resum

FulSynth és un sintetizador substractiu virtual. S'ha programat amb Pure Data i ha encapçalat tot un treball de documentació i recerca al voltant de sintetizadors substractius, les seves característiques i tècniques més frequents. S'exposaran els fonaments teòrics del mètod de síntesi substractiva i es comparará l'aplicació virtual programada amb tres sintetitzadors virtuals comercials de reconegut prestigi. Finalment, i com a fruit de tot el projecte portat a terme, s'extrauran les conclusions.

És imprescindible conèixer bé i entendre la síntesi substractiva per comprendre com funcionen molts dels dispositius o instruments electrònics que s'han vingut fabricant fins a dia d'avui. La majoria dels sintetizadors comercials, virtuals o analògics, usen la síntesis substractiva com a tècnica per produir i manipular el so.

El projecte desenvolupat ha requerit d'un aprenentatge previ en llenguatge Pure Data i porta finalment com a conseqüència la posada en pràctica i comprensió, tant creativa com tècnica, de l'ús de la síntesis substractiva com a eina de disseny i manipulació de sons.

Resumen

Fulsynth es un sintetizador substractivo virtual. Se ha programado con Pure Data y ha liderado todo el trabajo de documentación e investigación alrededor de los sintetizadores substractivos, sus características y técnicas más frecuentes. Se expondrán los fundamentos teóricos del método de síntesis substractiva y se comparará la aplicación programada con tres sintetizadores virtuales de reconocido prestigio. Finalmente, y como fruto de todo el proyecto llevado a cavo, se extraerán las conclusiones.

Es imprescindible conocer bien y entender la síntesis substractiva para comprender como funcionan muchos de los dispositivos o instrumentos electrónicos que se han fabricado hasta la actualidad. La mayoría de los sintetizadores comerciales, virtuales o analógicos, utilizan la síntesis substractiva como técnica para producir y manipular el sonido.

El proyecto desarrollado ha requerido un aprendizaje previo en lenguaje Pure Data y trae finalmente como consecuencia la puesta en práctica y comprensión, tanto creativa como técnica, del uso de la síntesis substractiva como técnica de producción y manipulación de sonidos. .

Acknowledgements

I was able to complete this exciting final project because of two people, supported by their respective institutions.

On one hand I'm grateful to my project supervisor, he has been truly hospitable and helpful from the very beginning of this almost one-year crusade. He gave me constant orientation and freedom to decide. I would like to thank Giuseppe for giving me the chance to come to the University of Limerick. I will always appreciate it. I would also like to thank DMARC. To learn more about your whole discipline will inspire me in the future.

On the other hand I want to sincerely thank my international coordinator in Barcelona, Jaume Comelles. He listened to my situation and he gave me the opportunity to come here trough the ERASMUS program. One more student abroad, I know, but one student fulfilled with his work. I want to mention as well Montse Targarona, for her constant feedback, work and support. I want to thank Katherine and the rest of the people from the University of Limerick International Office.

It is essential to acknowledge the Pd software developers, professionals in the audio field, recognised authors or simply enthusiasts I encountered while researching or gaining information in the Internet forums. They all deserve to be mentioned. They are an important part of my modest achievement.

Finally I want to mention the rest of my family, my two sisters Marta and Mònica, my aunts and my grandma. I love them and they helped me a lot, each one in their different sweet ways. I thank my old friends for being there when I needed them. This includes the people I've met here in Ireland. They all gave me smiles and courage in many situations. Thanks a million buddies!

List of Contents

Abstract i

Resum ii

Resumen iii

Acknowledgements iv

List of Contents v

List of Figures vii

CHAPTER 1. INTRODUCTION AND MOTIVATIONS 1

1.1. Reasons and purposes 1

1.2. Objectives 1

1.3. Quick Guide 2

1.4. Quick Introduction to Pure Data 3

CHAPTER 2. SUBTRACTIVE SYNTHESIS 4

2.1. Synthesis 4

2.2. Synthesizer 4

2.3. Hardware synthesizers 5

2.4. Analogue to digital 8

2.5. Software synthesizers 8

2.6. Subtractive synthesis 10

CHAPTER 3. FULSYNTH 24

3.1. Main interface 24

3.2. Block diagram 26

3.3. Oscillator 27

3.4. Antialiasing filter 29

3.5. ADSR Filter Bank 33

3.6. MOOG Filter Unit 35

3.7. Delay unit 38

3.8. NOISE Generator Unit 39

3.9. Low Frequency Oscillator (LFO) 39

3.10. Amplifier ADSR....41
3.11. Keyboard control....42

CHAPTER 4. TESTING AND COMPARISONS 44
4.1. Reference points 44
4.2. TEST1 - Waveshaping 45
4.3. TEST2 - Filtering....53
4.4. TEST3 – Amplification 58

CHAPTER 5. CONCLUSIONS AND FUTURE WORK 65
5.1. Theoretical conclusions....65
5.2. Practical conclusions 66
5.3. Future work 67

REFERENCE LIST 69

APPENDICES 1

Appendix A 1
1. FulSynth filters 2
2. Analysing FulSynth *svf*~ low-pass filter 2
3. Analysing the FulSynth *moog*~ low-pass filter 4

Appendix B 9
1. Starting with a synthesizer....10

Appendix C 11
2. Main Interface....12
3. ADSR modulated Filter Bank abstraction....17
4. MOOG Filter abstraction....24
5. DELAY Unit abstraction 25
6. NOISE Generator Unit abstraction 26
7. Low Frequency Oscillator Unit abstraction 27
8. Main Output and ADSR Envelope Generator Amplifier abstraction.29
9. Main Switch abstraction 29
10. Keyboard abstraction 30

List of Figures

Figure 1-I Pure Data example 3
Figure 2-I Moog Modular (1964) (Anonymous Blog [online]) 5
Figure 2-II Ladder filter topology (Radio-Electronics) 5
Figure 2-III Performance synthesizer: Minimoog (1970) (Wikipedia 2011)(Vidal Reynés 2000) 6
Figure 2-IV Roland sh3-a (Vintage Synth Explorer 2011) 6
Figure 2-V Aelita (URSS) (Matrixsynth 2010) 7
Figure 2-VI Novation Bass Station - Virtual Bass Synthesizer 8
Figure 2-VII Arturia Minimoog 9
Figure 2-VIII ABL Pro 10
Figure 2-IX Basics of subtractive synthesis 11
Figure 2-X Generic subtractive synthesizer block diagram 11
Figure 2-XI Sine wave 12
Figure 2-XII Square wave 12
Figure 2-XIII Sawtooth wave 13
Figure 2-XIV Triangle wave 13
Figure 2-XV Filters classification 15
Figure 2-XVI Ideal LPF 16
Figure 2-XVII Low-pass filter frequency response. fc at 3dB 16
Figure 2-XVIII White noise spectrum with content until 20Kz, limit of audible range 17
Figure 2-XIX Low-pass filtered noise. fc=1Kz 17
Figure 2-XX High-pass filtered noise. fc=5Hz 17
Figure 2-XXI Band-pass filtered noise. fc1= 300Hz; fc2=3400Hz. 18
Figure 2-XXII Ideal band-pass filter (Cassidy and Smith 2008) 18
Figure 2-XXIII Low-pass filter slope (eNotes 2011) 19
Figure 2-XXIV Angular frequency (eNotes 2011) 19
Figure 2-XXV Resonance (Q) in low-pass filters (DJ Jenkem Blog 2009) 20
Figure 2-XXVI Sound signal envelope (Schoenfelder 2009) 21
Figure 2-XXVII ADSR controlling frequency (ADSR (Filter) [online]) 22

Figure 2-XXVIII ADSR linear envelope generator (Making Music 2010).....22
Figure 2-XXIX ADSR exponential envelope generator (Peters) 23
Figure 2-XXX Basic subtractive ADSR synthesizer block diagram (Sievers).....23
Figure 3-I FulSynth: self-digital subtractive synthesizer implementation.....24
Figure 3-II Basic subtractive synthesizer block diagram.....25
Figure 3-III FulSynth: recognising the graphic interface 25
Figure 3-IV FulSynth complete block diagram 26
Figure 3-V FulSynth OSC1 and OSC2 abstraction 27
Figure 3-VI Waveshaping devlopement: sine, sawtooth, inverted sawtooth.....28
Figure 3-VII Waveshaping developement: trianle saw, triangle wave, square wave..28
Figure 3-VIII Waveshaping developement: pulse, rectangular wave.....28
Figure 3-IX FulSynth block diagram: from the source to antialiasing filter 29
Figure 3-X FulSynth: *antialiasing* filter abstraction (see appendix) 29
Figure 3-XI Nyquist frequency.....30
Figure 3-XII Nyquist graphic example.....30
Figure 3-XIII Ideal soundcard's antialiasing filter 30
Figure 3-XIV Audio input signal digitalized.....31
Figure 3-XV Aliasing 31
Figure 3-XVI FulSynth antialiasing filter, ideal situation 32
Figure 3-XVII FulSynth block diagram: from the source to ADSR Filter Bank.....33
Figure 3-XVIII FulSynth: ADSR filter bank abstraction 33
Figure 3-XIX FulSynth block diagram: from the source to MOOG Unit 34
Figure 3-XX FulSynth: MOOG Filter Unit abstraction.....35
Figure 3-XXI Initial FM frequency sine signal Figure 3-XXII Initial sine in positive values.....35
Figure 3-XXIII Setting AMP=2000 Figure 3-XXIV Setting DES=100 (offset) .36
Figure 3-XXV MOOG unit filter fc route.....36
Figure 3-XXVI MOOG unit frequency response (Samplecraze 2010) 36
Figure 3-XXVII FulSynth block diagram: from the source to Delay Unit.....37
Figure 3-XXVIII FulSynth: Delay unit abstraction 38
Figure 3-XXIX FulSynth: NOISE Generator Unit abstraction.....39
Figure 3-XXX FulSynth: Low Frequency Oscillator abstraction.....39
Figure 3-XXXI FulSynth complete block diagram 40

Figure 3-XXXII FulSynth: Amplifier and main output abstraction 41
Figure 3-XXXIII Source-modifiers. Subtractive synthesis block diagram 41
Figure 3-XXXIV Overdrive (Kreidler 2009).....41
Figure 3-XXXV FulSynth: keyboard abstraction 42
Figure 3-XXXVI Active keyboard keys (En el tintero de Zamzara 2006 Blog).....43
Figure 4-I Oscillator blocks in: Bass Station (top left) – ABL Pro (bottom left) – Minimoog (right) 45
Figure 4-II Oscillators for each synthesizer - Bass Station (top) - Minimoog (middle) - ABL Pro (bottom).....46
Figure 4-III Novation Bass Station LOW (left) and HIGH (right) frequency square wave 47
Figure 4-IV Novation Bass Station MID frequency square wave 47
Figure 4-V Arturia Minimoog LOW (left) and HIGH (right) frequency square wave47
Figure 4-VI Arturia Minimoog MID frequency square wave 48
Figure 4-VII ABL Pro LOW (left) and HIGH (right) frequency square wave.....48
Figure 4-VIII ABL Pro MID frequency square wave.....49
Figure 4-IX FulSynth LOW (left) and HIGH (right) frequency square wave.....49
Figure 4-X MID frequency square wave 49
Figure 4-XI Novation Bass Station Oscillators 51
Figure 4-XII FulSynth oscillators 52
Figure 4-XIII Bass Station oscillator modulation controls 52
Figure 4-XIV Filter Blocks in Bass Station (top left) – Minimoog (top right) – ABL Pro (bottom) 53
Figure 4-XV Situation of the filter for each synthesizer- Bass Station (top)- Minimoog (middle)-ABL Pro (bottom).....54
Figure 4-XVI Novation Bass Station LOW (left) and HIGH (right) filter cut-off frequency.....55
Figure 4-XVII Arturia Minimoog LOW (left) and HIGH (right) filter cut-off frequency.....55
Figure 4-XVIII ABL Pro LOW (left) and HIGH (right) filter cut-off frequency.....55
Figure 4-XIX FulSynth LOW (left) and HIGH (right) filter cut-off frequency 56
Figure 4-XX Arturia Minimoog filter.....57
Figure 4-XXI Arturia Minimoog filter's main parameters 57

Figure 4-XXII ADS Minimoog envelope generator 58
Figure 4-XXIII Amplifier in each synthesizer- Bass Station (top)- Minimoog (middle)-ABL Pro (bottom) 59
Figure 4-XXIV Novation Bass Station - Amplitude variation depending on frequency 60
Figure 4-XXV Arturia Minimoog - Amplitude variation depending on frequency 60
Figure 4-XXVI AudioRealism Bass Line Pro - Amplitude variation depending on frequency 61
Figure 4-XXVII FulSynth - Amplitude variation depending on frequency 61
Figure 4-XXVIII Human ear frequency response 62
Figure 4-XXIX ABL Pro amplification panel 63
Figure 5-I FulSynth block diagram remade 65
Figure 5-II FulSynth filters 2
Figure 5-III Impulse Pd generated 3
Figure 5-IV Pd *svf~* low-pass filter frequency response. Fc= 140Hz (top). Fc=1100Hz (bottom) 3
Figure 5-V Pd *svf~* low-pass filter frequency response. Fc= 1,11kHz 4
Figure 5-VI Pd *moog~* filter frequency response. Fc=900Hz (top). Fc=1400Hz (bottom) 5
Figure 5-VII FulSynth MOOG Unit 6
Figure 5-VIII MOOG Unit values 6
Figure 5-IX MOOG Unit cut-off frequency (fc) modulation 6
Figure 5-X Peak values for the moog~ filter frequency response 7
Figure 5-XI Pd *moog~* filter cut-off frequency modulated by a 3Hz low frequency oscillato 7

CHAPTER 1. INTRODUCTION AND MOTIVATIONS

1.1. Reasons and purposes

Developing a subtractive synthesizer using Pure Data (Pd) (Puckette and Pd community 1996, Wikipedia 2009) will allow one to understand how many manipulating sound devices work. Although nowadays most of devices combine other techniques and tools, researching subtractive synthesis and subtractive synthesizers is a good introduction to the Music Technology field.

Beyond an understanding of all the processes, circuitry or techniques that might be involved, there exists an artistic and musical interest. Using computer science and electronics awareness, the design process will take into consideration the user's sense of application. It is essential for good training in the field of Music Technology to understand how one translates signal into sounds, filtering into music, noise into harmony or oscillators into rhythm. It is necessary to understand the Music part to make an efficient use of the engineering, i.e. Technology.

The market is huge and instruments, devices, synthesizers, vocoders, workstations, etc. are designed for many different reasons and purposes. The self-design application has a modest purpose: to be a standalone instrument, a synthesizer which one could feel comfortable enough to experiment with, emulating sounds or being able to play some standard electronic music live. A synthesizer based on subtractive synthesis theory will be built to understand the common control parameters this kind of instrument and many others might incorporate and to appreciate the expressive possibilities of the subtractive synthesis method.

1.2. Objectives

There is little point in comparing this project with other mature and commercial virtual synthesizers. It will not be competitive. To learn is the only possible justification one beginner Pd programmer could claim to have.

The main objective of this final project is to understand subtractive synthesis as an engineer, its background and features, emulating an analogue virtual synthesizer with Pd; reading the method's features as a musician, viewing its sense of sound design application, utility or expressive possibility.

The research field: synthesizer devices, its electronics, main features, sounds, tools and techniques.

1.3. Quick Guide

Chapter 2: Subtractive Synthesis is going to set out the fundamentals of this sound synthesis method. The synthesizer is going to be introduced as a musical instrument. The technical concepts introduced in this chapter presuppose that the reader has a minimum background in physics of sound, signal theory, mathematics and engineering.

Chapter 3: FulSynth introduces the self-developed application. Each single part of it is going to be explained without going deep into Pure Data theory. The reader will be able to understand FulSynth main functionalities and purposes.

Chapter 4: Testing and Comparisons will test and compare FulSynth with other virtual synthesizer software. Preliminary conclusions will be extracted from the observation and the comparison with the expected theoretical values.

Chapter 5: Conclusions and Future Work will finally discuss the utility of the self-developed application, setting its advantages and limitations, taking into consideration the whole project and its researching work. Future Work will set out the first steps to follow in order to improve the obtained results.

Appendices A and B introduce additional information about FulSynth in order to fulfil a thirsty reader. *Appendix C* prints FulSynth's most relevant Pd self-designed blocks and functions.

1.4. Quick Introduction to Pure Data

The Pure Data (Pd) (Puckette 1996) community is based on the Internet at the URL address puredata.info. The online portal introduces its first-time visitors to a useful definition:

> *Pd (aka Pure Data) is a real-time graphical programming environment for audio, video, and graphical processing. It is the third major branch of the family of patcher programming languages known as Max (Max/FTS, ISPW Max, Max/MSP, jMax, etc.) originally developed by Miller Puckette and company at IRCAM. The core of Pd is written and maintained by Miller Puckette and includes the work of many developers, making the whole package very much a community effort.*

Pure Data is a free open-source high-level programming language. Its software is useful to develop audio or video applications. It works using an object-connection schematic methodology on a white plain interface. It incorporates different libraries filled with default objects, which are functions already programmed (with a lower-level language) by Pd-enthusiasts developers. The user programs using these objects and understanding Pd rules, concepts and tools (signal flow, messages, atom boxes, symbols, etc.). One is able to create, depending on his mathematical and technical skills, a sound or graphical based computer application.

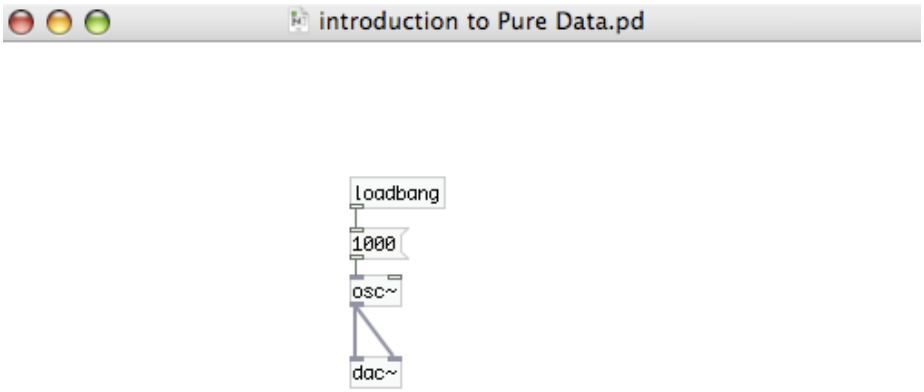


Figure 1-I Pure Data example

When *introduction to Pure Data.pd* is first loaded 1kHz tone is sent straight away to the computer soundcard.

CHAPTER 2. SUBTRACTIVE SYNTHESIS

2.1. Synthesis

There are many different types of synthesis and the trend to consider a technical meaning easily predominates over a common definition. Cambridge dictionary (Cambridge, 2010) defines synthesis as:

> *"the mixing of different ideas, influences or things to make a whole which is different or new".*

The process of synthesis itself is thus a bringing together and it is necessary to put emphasis on *"to make a whole"*: more than assembling randomly, synthesis should be a creative process. It is this artistic aspect that is often overlooked in favour of the more technical aspects of the subject.

Choosing amongst the different existing types of synthesis this report is interested only in the one used to create music and sounds.

2.2. Synthesizer

All musical instruments can be thought of as being synthesizers; sound synthesis is not only concerned with sophisticated computer-generated timbres. Sound synthesis is the process of producing sound and sound might be produced mechanically, electronically, using mathematics, physics or even biology. However, the word synthesizer has recently come to mean exclusively an electronic instrument capable of producing a wide range of different sounds.

Taking this current consideration, it is possible to divide a sound synthesizer in two basic functional blocks: a *control interface*, where the parameters that define the end product are set by the user and a *synthesis engine*, which interprets the parameter values and produces the output.

Most of the commercial synthesizers can be classified inside a *performance synthesizer* group, in which their synthesis engine is barely modifiable and their internal modules are already built-in. This enables the rapid patching of the most

commonly used configurations. However, there is a second big group to be mentioned: the *modular synthesizers,* which despite their slow patch configuration and consequently long set up time have much greater flexibility and sound possibilities.

2.3. Hardware synthesizers

Figure 2-I Moog Modular (1964) (Anonymous Blog [online])

Dr. Robert Moog (May 23, 1934 – August 21, 2005) (Wikipedia 2011) created and patented a new way to introduce filters in music-oriented analogue synthesizers. He designed a low-pass filter that allowed to pass some distortion other builders were avoiding. He created an innovative sound that has brought him prestige and commercial success until today.

The classical analogue Moog filter is a 24 dB-per-octave voltage controlled low-pass (904A model) (filtering section: 2.3.5). It is usually said it is a "ladder" low-pass filter, which defines the topology he chose, the ladder or Cauer (Wikipedia 2010) filter topology.

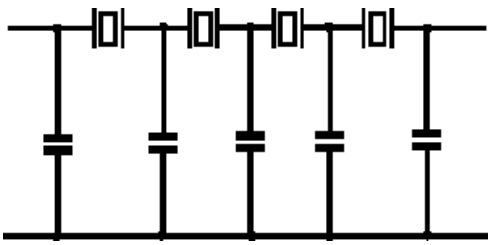


Figure 2-II Ladder filter topology (Radio-Electronics)

It is possible to find the patent and the schematics of the classical Moog filter (Tillman 2006). Although its full schematic is not going to be analyzed, electronic

analogue filters deserved here at least a wink among this whole subtractive synthesis dissertation.

The modular Moog (Figure 2-I) gave the performer great flexibility but it was with the compact Minimoog version (Figure 2-III) released by R.A. Moog Inc. company (U.S.A., 1970) (Wikipedia 2011), that huge commercial success was achieved.

Figure 2-III Performance synthesizer: Minimoog (1970) (Wikipedia 2011)(Vidal Reynés 2000)

The user interface designed in many performance-oriented synthesizers is quite similar in different models have been built over the years. It has a keyboard and a parameter control area. Roland sh3-a (1974) was the first analogue monophonic synthesizer this brand created:

Figure 2-IV Roland sh3-a (Vintage Synth Explorer 2011)

It included subtractive and additive synthesis. Bands such as The Human League or Blondie were using it (Wikipedia 2011).

Built by the Murom radio plant in the old USSR (Matrixsynth 2010) there is the vintage Aelita analogue monophonic synthesizer. It has three oscillators:

Figure 2-V Aelita (URSS) (Matrixsynth 2010)

2.3.1. Monophonic synthesizers

Hardware performance-oriented synthesizers appeared as monophonic instruments: instruments designed to play melodies, solos or lead lines. However, polyphony (the possibility of playing more than one key at once) was rapidly incorporated as a switchable option in many devices. Monophonic synthesizers appeared in the analogue times, but they are still used by many musicians or studied and virtually implemented by some software development companies.

Most of monophonic subtractive synthesizers include tools or options that are not needed to understand the subtractive synthesis, but these tools definitely improve the instrument's features. It is common to find a monophonic synthesizer including for instance two or more oscillators -which can be configured at different octaves to compensate a small keyboard span-, a bank of filters, ADSR envelope generators, a delay unit, an arpegiator unit, a low frequency oscillator, etc., among many other sound shaping options. The following chapters will explain or introduce some of these techniques or features. Although some of these tools have been modestly implemented with Pd into the self-developed software, the study of each of the techniques and tools a monophonic synthesizer may include is not the objective of this dissertation.

2.4. Analogue to digital

Many considerations and terminology that are used today among the vast diversity of digital synthesizers might come from the past. For instance, widespread names as voltage-controlled oscillators (VCO) come from the analogue era and are used to designate the analogous concept in digital devices. Subtractive synthesis used in analogue synthesizers was an analogue method itself that today digital synthesizers try to emulate. First subtractive synthesizers were analogue. There are many terms and techniques mixed up between analogue and digital systems and this served to improve the familiarity to anyone who already knew about analogue synthesizers and their processing techniques, but it has brought confusion to young musicians without a strong background.

It is not the aim of this project to go deep inside the differences between analogue and digital signals and processes. Although from the inside the rules of the game may differ, to understand how subtractive synthesis works (in a digital implementation of an analogue subtractive synthesizer) it is not required to follow all the steps of the discretization, quantification and sampling chain. The final results will provide the necessary answers. Analogue to Digital Converters and Digital to Analogue, in this case, will not be investigated. Specifications will be provided when required.

2.5. Software synthesizers

2.5.1. Novation Bass Station

Figure 2-VI Novation Bass Station - Virtual Bass Synthesizer

Novation launched the original hardware monophonic subtractive synthesizer Bass Station in 1993. Nowadays its digital emulation, the virtual Bass Station, can be purchased and used as a virtual instrument (VST) or audio unit (AU) plug-in (Novation 2007).

2.5.2. Arturia Minimoog

Figure 2-VII Arturia Minimoog

Arturia succesfully emulates the classical Minimoog analogue subtractive synthesizer. It can be purchased and played either as a standalone instrument or as a plug-in (Arturia 2009).

2.5.3. AudioRealism Bass Line Pro (ABL Pro)

Figure 2-VIII ABL Pro

AudioRealism developed Bass Line Pro, an interesting subtractive synthesizer with an integrated pattern sequencer. It can be purchased and used as a plug-in (Realism 2009).

Novation Bass Station, Arturia Minimoog and AudioRealism Bass Line Pro will be deeply analysed in Chapter 4.

2.6. Subtractive synthesis

2.6.1. Definition

In general terms, the subtractive synthesis technique is based around the idea that all musical instruments can be broken down into three major parts: a *source* of sound, a *modifier* processing the source output and some *controllers,* used as a performer-instrument interface. Although this model is powerful for helping to understand how musical instruments work, it gives a simplified answer. However, the idea of modifying the output of a sound source is useful to describe how subtractive synthesis works.

Subtractive synthesis can be split into two main concepts: *source* and *modifier*. The source is able to produce sounds with different harmonic-content. The modifier filters out any unwanted harmonics and shapes the sound's final volume envelope. The filter thus 'subtracts' the unwanted frequencies, hence the name of the synthesis method.

In its most basic form, subtractive synthesis technique follows (Abdullah Blog [online]):

OSCILLATOR → FILTER → AMPLIFIER

Figure 2-IX Basics of subtractive synthesis

An oscillator generates a "suitable sound" that is routed through a filter. The filter erases or cuts-down the undesired frequencies giving the sound any chosen timbre. Afterwards the resulting sound is routed to the amplifier stage. This one controls the loudness of the final sound.

A sine wave would not be a "suitable sound" source. It has just one harmonic. If it were subtracted there would be no sound. A "suitable sound" needs a minimum harmonic richness. One triangle wave has few harmonics as well. However, it is common to combine different signals before the filter stage. Combining a sine wave with a triangle and a square the resulting added wave would have enough harmonics to subtract. A new sound could be created now by subtraction. An initial block diagram:

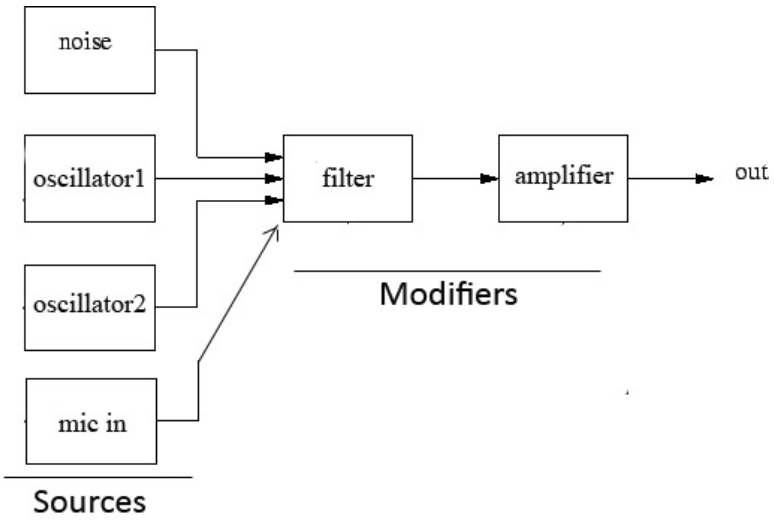


Figure 2-X Generic subtractive synthesizer block diagram

2.6.2. Sources

The common sound sources used in subtractive synthesis synthesizers are based on mathematics. There are two basic types: *waveforms* and *random.* A waveform is the shape the oscillator generates while a random sound or wave produces *noise*, which contains a changing mixture of all frequencies.

The typically used waveforms are: sawtooth, square, pulse, sine and triangle, although it is also possible to find rectangular waves (square wave with a different duty cycle and different harmonic content) or inverted sawtooths. These waves are easily represented. Each one of them has a different spectrum, which provide colour, brightness or timbre possibilities to the synthesizer.

These are the mathematical representations in time and frequency domain of some common waveforms (John Hopkins University and Ross 2010):

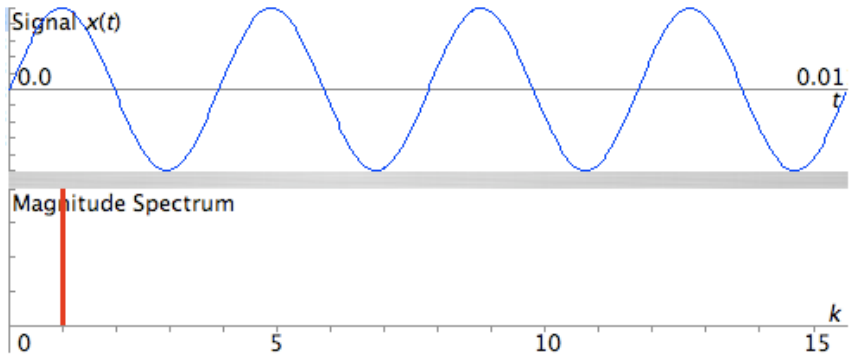


Figure 2-XI Sine wave

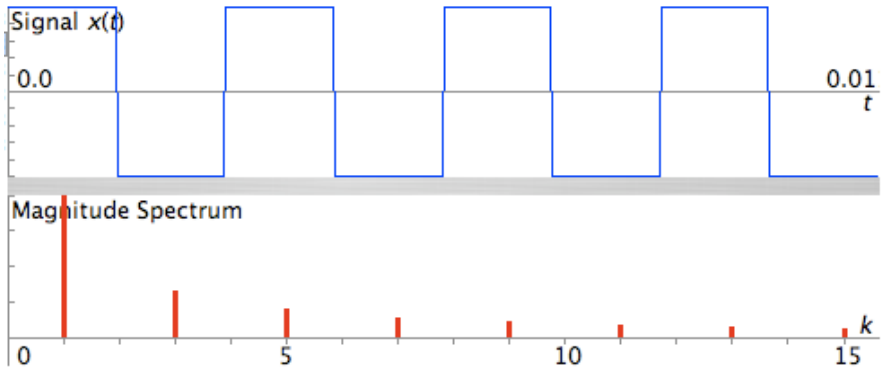


Figure 2-XII Square wave

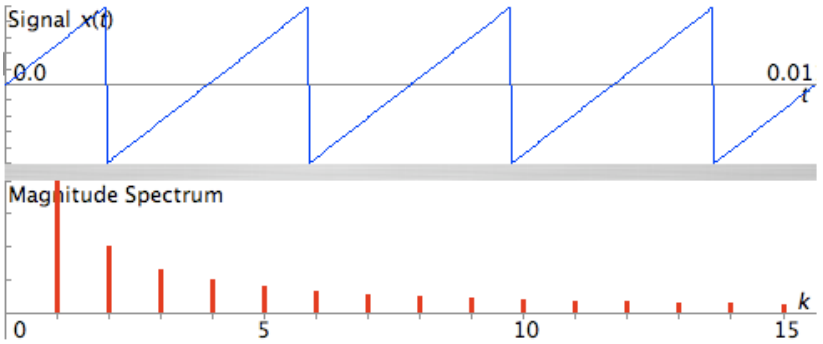


Figure 2-XIII Sawtooth wave

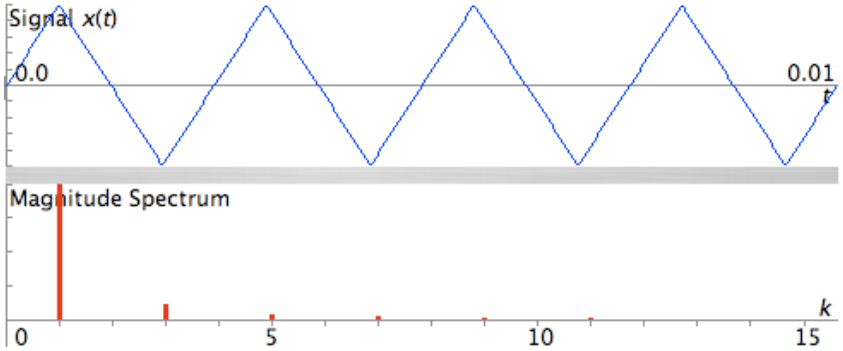


Figure 2-XIV Triangle wave

The frequency domain shows exactly how much it is possible to subtract. The *sine* wave is a pure tone. It has just one harmonic, its fundamental. It makes a smooth sound. It might be possible to combine within other tools or options a synthesizer may offer. The *triangle* wave has two main linear slopes and a small amount of odd-numbered partials, which it gives a bit more harmonic content for a filter to work on. Its sound is smooth also, but it tends to have the synthetic feel of a sawtooth or square wave. The *square* wave is unique because it contains only odd multiples of the fundamental harmonic and has a “hollow” sound. The *sawtooth* wave contains both odd and even harmonics and sounds tough and bright.

To manipulate the harmonic content of any sound is to change its expressive characteristics without it being necessary to change its pitch or volume. For instance, playing a 440Hz sine wave with a synthesizer, a guitar or a piano will give the same note with a different timbre (different harmonic content). Subtractive synthesis whether analogue or digitally manipulates through filtering the harmonic content of mathematical signal waves. The waves created by the oscillators, function generators, are sent to the filter stage.

There could be different physical ways to trigger and control the audio waves from the source. However, subtractive synthesizers use a piano keyboard controller. This sets the relation between a wide possibility of pitches and one typical chromatic distribution. It is a link between common music and experimentation and it provides an efficient controller to live performances. It allows communication with other musicians. When a key is pressed the synthesizer triggers the oscillators with the chosen waves, filtering and amplification.

2.6.3. Modifiers: Introduction

The classic subtractive synthesizers sound shapers or modifiers can be split in two major groups: *filters* and *amplifiers*. As it has already been mentioned, filtering is used to change the harmonic content or timbre of the sound; amplification is used to change the volume or "shape" of it. *Envelope generators* (EG) might control both types of modifiers through control voltages (CV) [analogue devices anachronism found in many digital context]. Envelope generators permit to modify the sound into typically four different time subdivisions.

2.6.4. Modifiers: Filters

Filters are a key element defining the personality of any synthesizer. They modify the signal source frequency contour. Nowadays there are many others digital sound manipulation techniques, however, in the earlier days, filters characterized the sound of a synthesizer more than anything else. Filters are powerful timbre modifiers. They change the relative proportions of harmonics in a sound. They subtract the unwanted part of a metaphorical shapeless rock in a sound sculpture process. Vintage synthesizers such as the Minimoog are priced more than others because of the distinctive sound of their filters.

A filter is an amplifier whose gain changes with frequency, but because of a recurrent convention that fixes their maximum gain to one, it is more correct to say it is its attenuation that changes with frequency.

One common classification method for filters is based on the shape of the gain curve, i.e. the curve of its frequency response module. The typical frequency response curves are (BMARS Blog [online]):

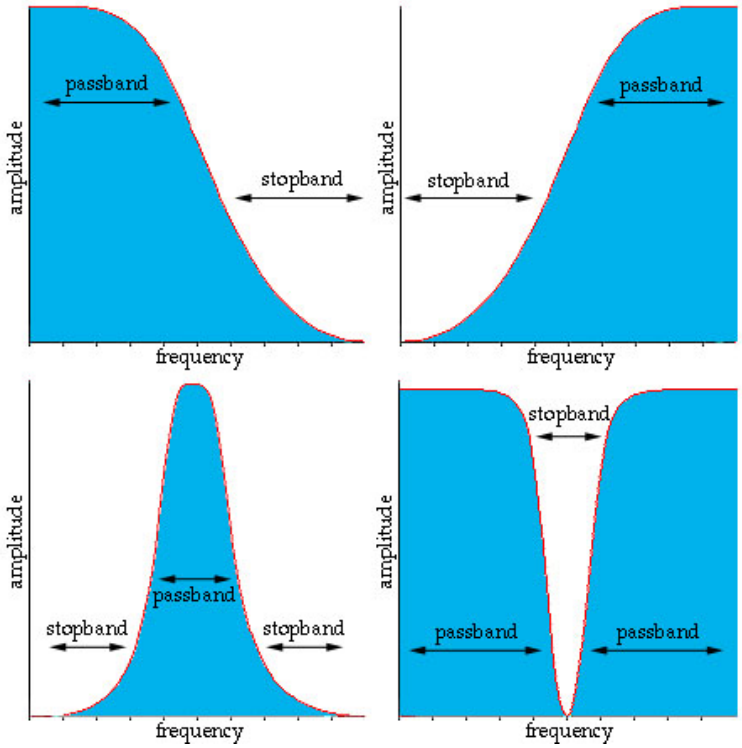


Figure 2-XV Filters classification

Its names: *low-pass, high-pass, band-pass, notch.*

A low-pass filter (LPF) attenuates more of the input sound signal as the frequency increases; letting the low frequencies pass. A high-pass (HPF) does exactly the opposite, attenuates the lowest frequencies and lets the high ones pass. The band-pass filter (BPF) chooses a band of the spectrum and attenuates the rest and the notch filter acts in the opposite way.

Filters in subtractive synthesizers always include two control parameters: the *cut-off frequency* and the *resonance* parameter. Sometimes it is also possible to set the *slope* of the filter frequency response.

Cut-off frequency:

The cut-off frequency is the frequency at which the attenuation is 3dB. At this point the attenuation first becomes apparent. Representing an ideal frequency response of a low-pass filter illustrates the concept (Gentry and Knipmeyer 2008):

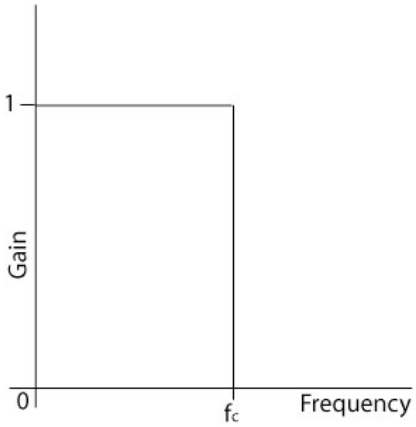


Figure 2-XVI Ideal LPF

At fc half of the power in the audio signal has been lost. Below fc, a low-pass filter has no effect on the audio signal.

Closer to a real curve:

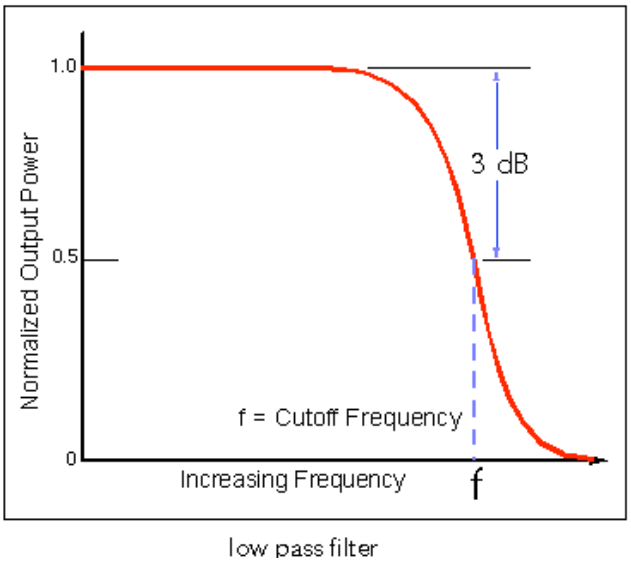


Figure 2-XVII Low-pass filter frequency response. fc at 3dB

The cut-off frequency can be controlled from the synthesizer user interface and gives the possibility of windowing the part of the spectrum we want. In other words, it subtracts an undesirable harmonic spectrum.

Representation and importance of the cut-off frequency: An ideal white noise spectrum filtered by different ideal filters, represented on a logarithmic frequency axis (John Hopkins University and Ross 2010):

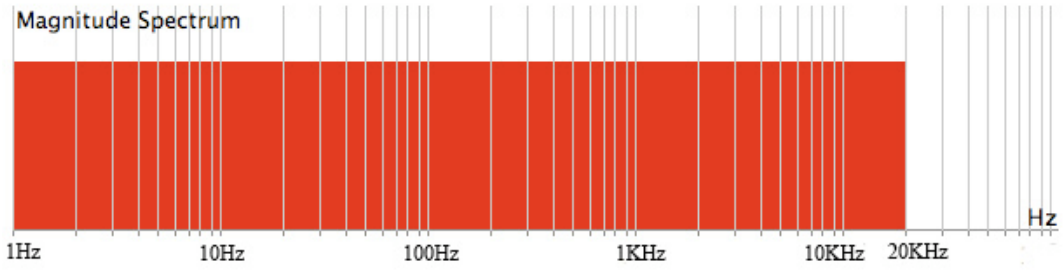


Figure 2-XVIII White noise spectrum with content until 20Kz, limit of audible range

The action of different filters:

Low-pass filter (LPF):

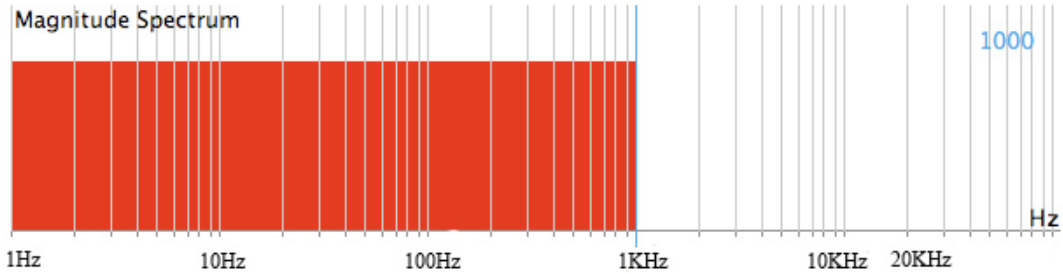


Figure 2-XIX Low-pass filtered noise. fc=1Kz

High-pass filter (HPF):

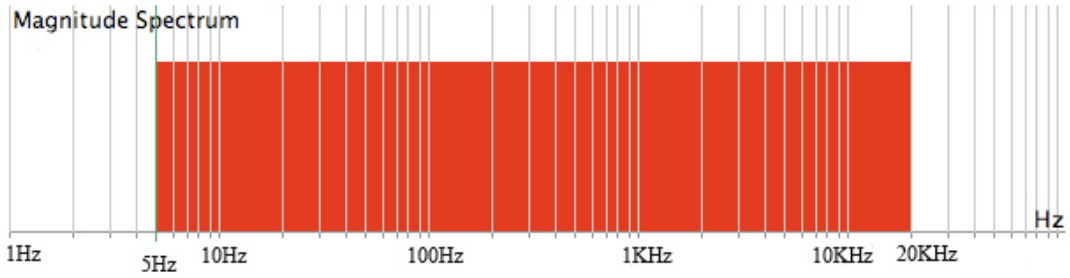


Figure 2-XX High-pass filtered noise. fc=5Hz

Comment: There is no music performance significance in the use of this windowing. However, it is possible to use this filter to design or program a synthesizer. A high-pass filter with fc=5Hz cuts the harmonic content at f=0Hz. This filter might be useful to avoid an unwanted signal offset.

Band-pass filter (BPF):

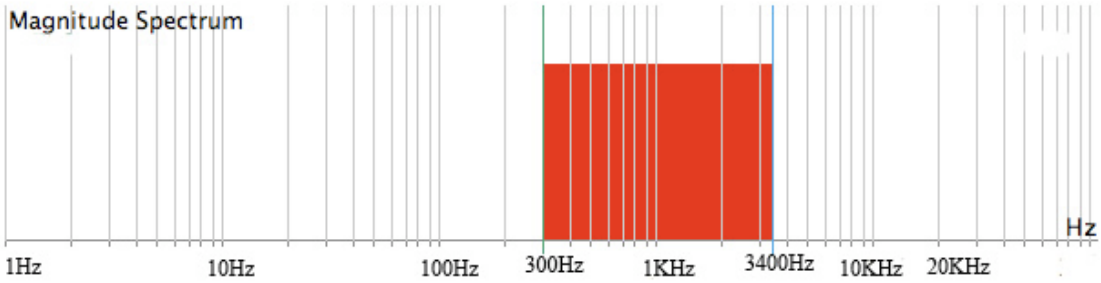


Figure 2-XXI Band-pass filtered noise. fc1= 300Hz; fc2=3400Hz.

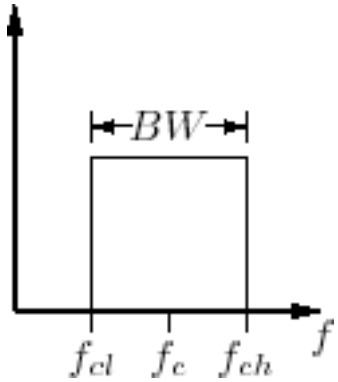


Figure 2-XXII Ideal band-pass filter (Cassidy and Smith 2008)

The cut-off frequency is placed in the middle of the band-pass. To change this parameter in a synthesizer drags a band or window along the frequency spectrum.

Comment: Figure 2-XXI. The effect of passing a human voice through that filter is known as the "telephone filter effect". This useful band allows saving bandwidth in communications. This might provide a tool to the audio-performer; it creates a well-known effect to the listener.

The slope:

Some synthesizers may also include the possibility to choose among different attenuation curves for each kind of filter. The sharpness of the curve slope determines how flexible or exact a filter is. In terms of analogue or digital electronic design, this determines the increase in cost and also the computational effort a device requires.

In technical terms, simple filters (one pole) have slopes of 6dB/octave, which means that for each doubling of frequency the attenuation increases by 6dB. Two-pole

filters will have an attenuation of 12dB/octave and a four-pole filter will have an attenuation of 24dB/octave. The slope increases as the number of poles increase. A two-pole filter is usually associated with a more natural sound while a four-pole one has more a synthetic tone and makes much larger changes to the timbre as the cut-off frequency is changed.

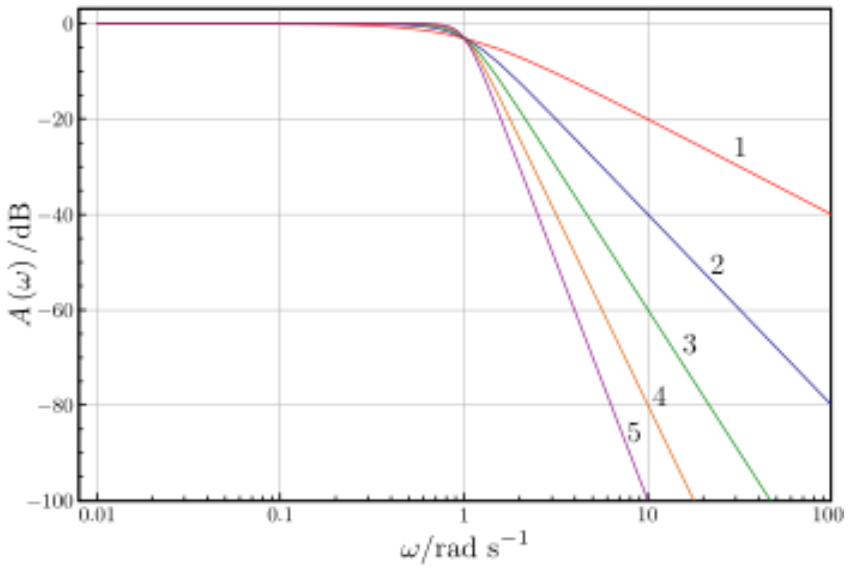


Figure 2-XXIII Low-pass filter slope (eNotes 2011)

$$w = 2*\Pi*f$$

Figure 2-XXIV Angular frequency (eNotes 2011)

From order one to order five the slope sharpness increases.

Resonance:

This is a peaking or accentuation of the filter frequency response (gain) at a specific frequency. For synthesizer low-pass filters and high-pass the resonance is usually placed at the cut-off frequency. For the band-pass filter the resonance is given by the formula: *Q= fc/ Bandwidth* (Russ 1996), the peak is placed at fc and it is related to the pass-band: it decreases its accentuation if the band (*Bandwidth*) gets wider.

Low-pass filter frequency response with and without resonance:

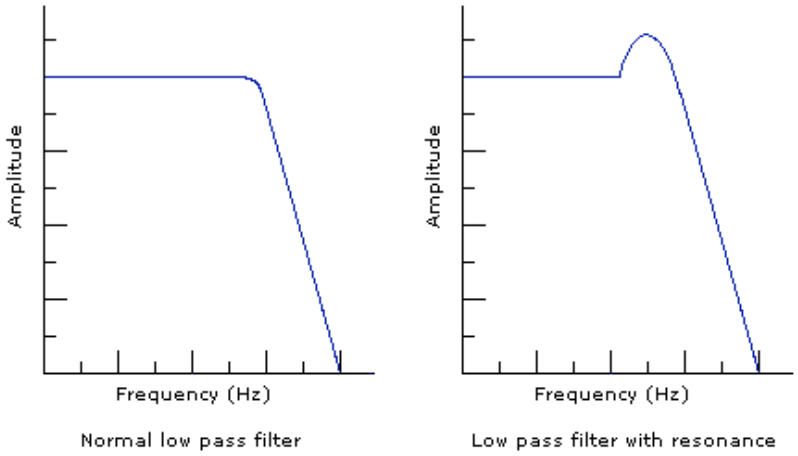


Figure 2-XXV Resonance (Q) in low-pass filters (DJ Jenkem Blog 2009)

2.6.5. Modifiers: Amplifiers

In a subtractive synthesizer there is an amplifier as a second modifier, it modifies the volume, gain or amplitude of the signal coming out from the filter output (Figure 2-IV).

The amplitude of a signal is related to the perceived loudness and the dynamics of the performance (however, in a mathematically inexact way (Puckette 2006)). An amplifier input is typically connected to a master volume and an envelope generator, which helps to emulate the dynamics a natural instrument may offer. While a filter unit modifies the harmony and melody of a composition, the amplifier stage may give a *crescendo*, *piano, forte,* etc. The performer will consider the importance of the dynamics in a musical composition when required.

Comment: As an anachronism from the analogue era it is still common to find the amplifier unit is called a VCA unit (voltage-controlled amplifier) even if the device is a digital emulation.

2.6.6. Modifiers: Envelope generator

The envelope of a signal is its outline, the contour of the signal. In communications or signal processing theory the envelope of a signal might carry an important information content that an envelope detector would translate.

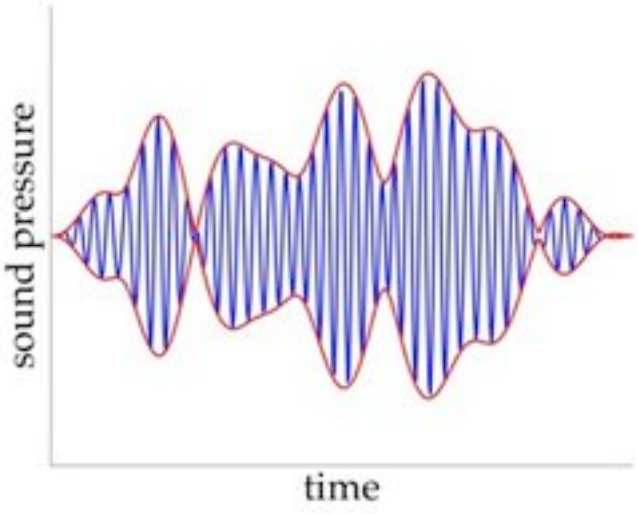


Figure 2-XXVI Sound signal envelope (Schoenfelder 2009)

Envelope generators appear often as controllers in the filtering and amplifier stage of a subtractive synthesizer. It is common for them to modulate the cut-off frequency of the filter and shape the final sound envelope in the amplifier stage.

Envelope generators control frequency (filter stage) or volume (amplifier) range as a function of time. The values are controlled into differentiated time segments, commonly four: *Attack time,* controls the length of the time a value takes from zero until a higher level; immediately after starts the *decay time,* the time the values take now to decrease until a constant level. Values remain steady at the same constant level during the length of the *sustain time*, and the time values taken to decrease until the initial reference level is called *release time*. This four-segment envelope generator is called ADSR envelope generator and it is triggered or released depending on the press/release action of the synthesizer keyboard key (see Figure 2-XXII).

Envelope generator controlling frequency values (filter stage):

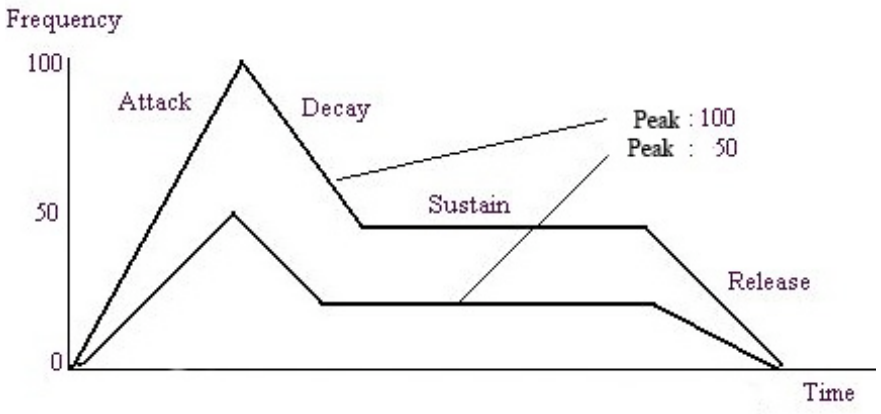


Figure 2-XXVII ADSR controlling frequency (ADSR (Filter) [online])

The value of the cut-off frequency of a filter may change as shown in Figure 2-XXVII. Having a cut-off frequency set at 0Hz the envelope generator adds to it a value between 0 and the *Peak* level. There is thus a frequency deviation around the cut-off frequency defined over a certain time period (see *modulation,* section 3.6.1).

Envelope generator controlling volume values (amplifier stage):

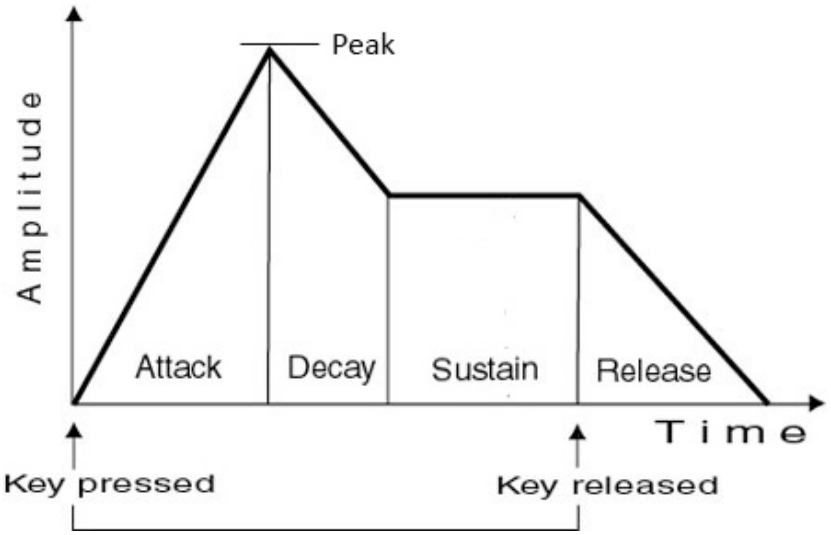


Figure 2-XXVIII ADSR linear envelope generator (Making Music 2010)

The sound volume (Amplitude) values are controlled through this function (Figure 2-XXII). From an initial volume 0 (silent) the sound will reach the *Peak* amplitude value set by the user during the attack time.

Comment: One bowed instrument may have a longer attack time in its volume than for instance a piano or a drum.

As many real-world sounds change in non-linear way, it is common to find synthesizers with an exponential envelope generator option, which provides sometimes a better approach to create more realistic or natural sounds. Furthermore, *"much has been made of the supposedly logarithmic nature of human hearing"* (Puckette 2006). The logarithmic scale the human ear uses might explain why an exponential envelope generator sounds more natural to us.

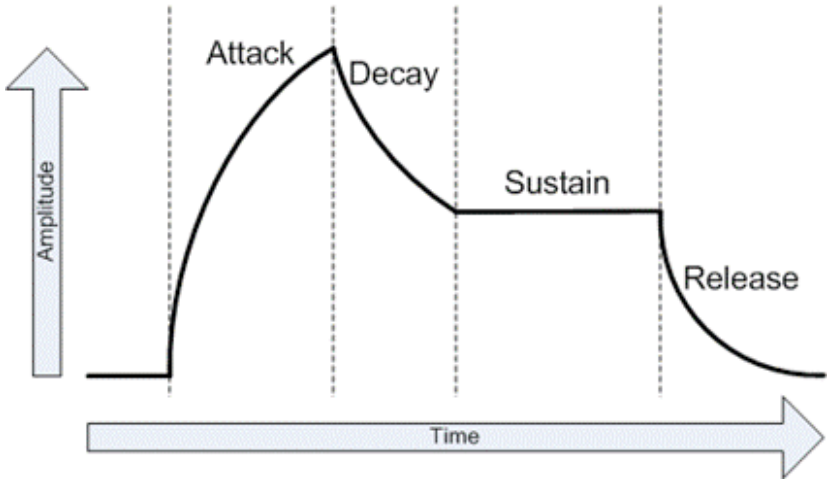


Figure 2-XXIX ADSR exponential envelope generator (Peters)

Block diagram of a basic subtractive synthesizer:

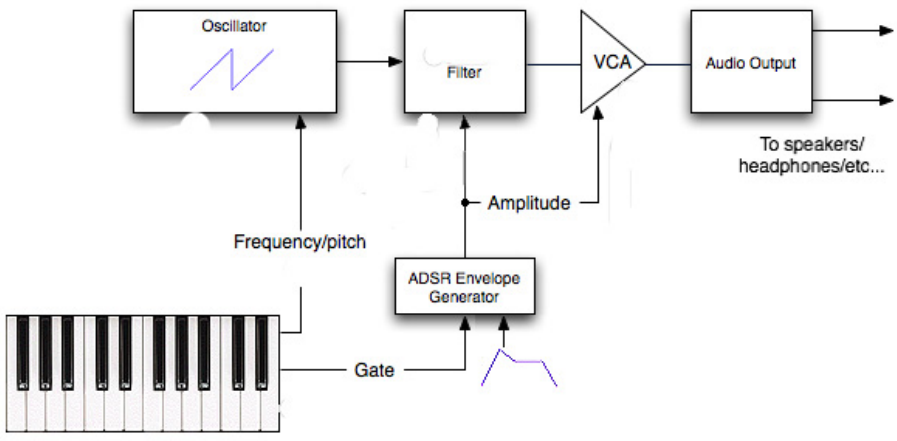


Figure 2-XXX Basic subtractive ADSR synthesizer block diagram (Sievers)

CHAPTER 3. FULSYNTH

3.1. Main interface

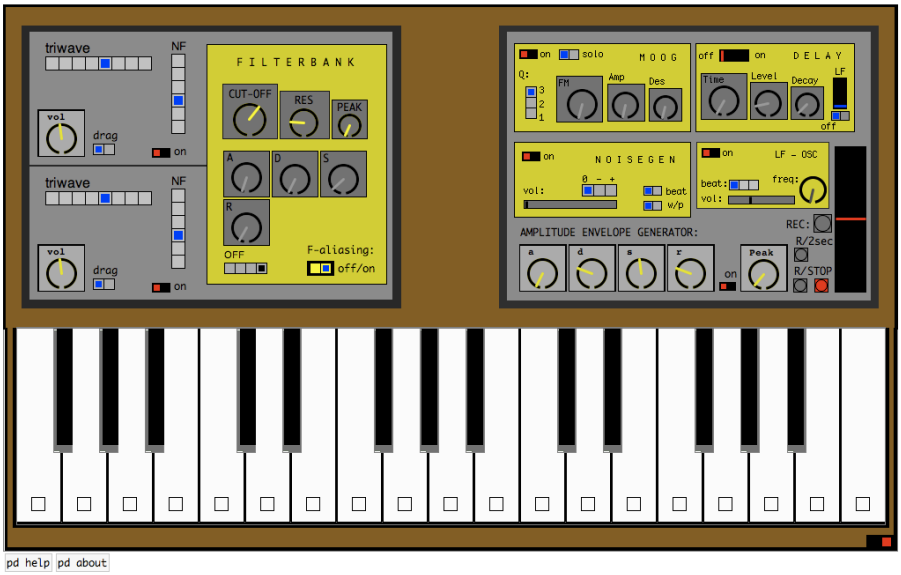


Figure 3-I FulSynth: self-digital subtractive synthesizer implementation

This software has been fully developed with Pd (Puckette 1996), using its standard functions and components. Arturia Minimoog and other commercial or open source virtual synthesizers have led the researching and the learning process. Its name is FulSynth and the Figure 3-I shows its main interface.

The main graphic interface is designed observing the patterns most monophonic synthesizers follow. As many of them do, the oscillator section (main signal source) is placed at the top-left corner and precedes the filter bank section. At the right side the ADSR amplifier stage ends the chain. The keyboard triggers the sources. Although FulSynth's block diagram is a bit more complex than the basic subtractive synthesizer block diagram already shown (Figure 2-X, section 2.6.1) it follows the same logic:

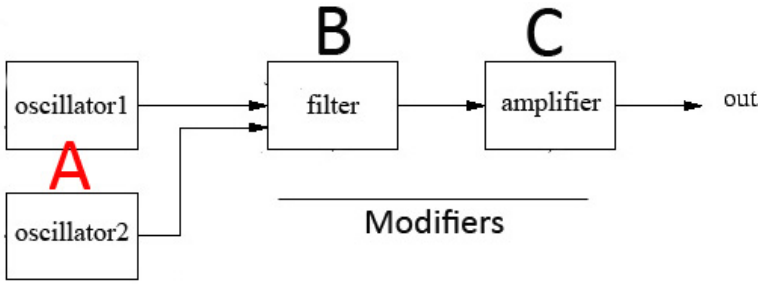


Figure 3-II Basic subtractive synthesizer block diagram

The ‘A’, ‘B’ and ‘C’ blocks placed at FulSynth graphic interface:

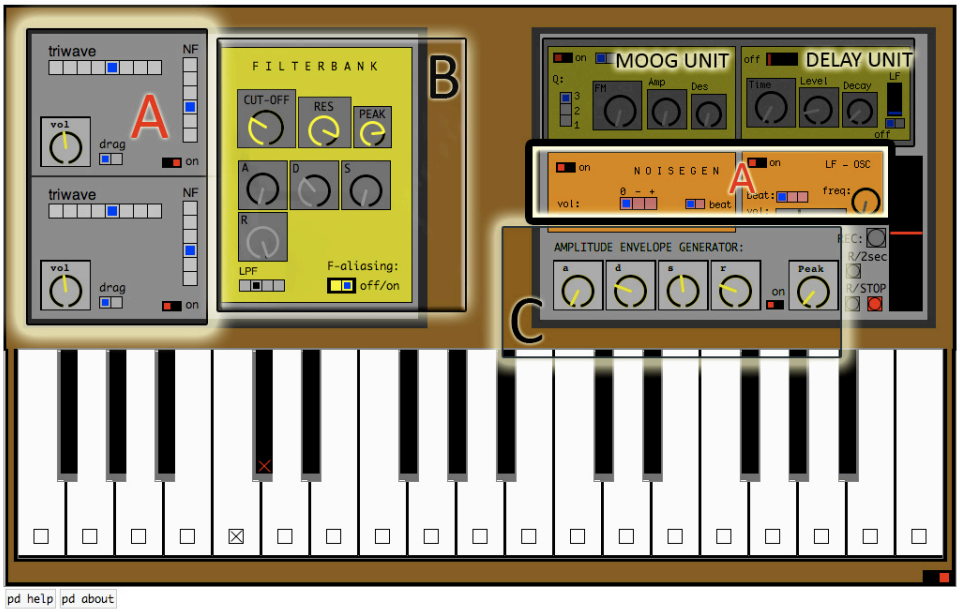


Figure 3-III FulSynth: recognising the graphic interface

The main signal sources (‘A’ on the left side, Figure 3-III) are the oscillator banks. However, there are two additional sound sources placed at the right side (‘A’ on the right side, Figure 3-III). These are the NOISE Generator and the Low Frequency Oscillator (LFO). The ‘B’ block is the filtering stage. It is also possible to incorporate a MOOG Filter Unit and a Delay Unit before the ‘C’ block. This last one incorporates a master volume control and the already explained ADSR envelope controllers (section 2.6.6). The next section will show the FulSynth block diagram in detail.

3.2. Block diagram

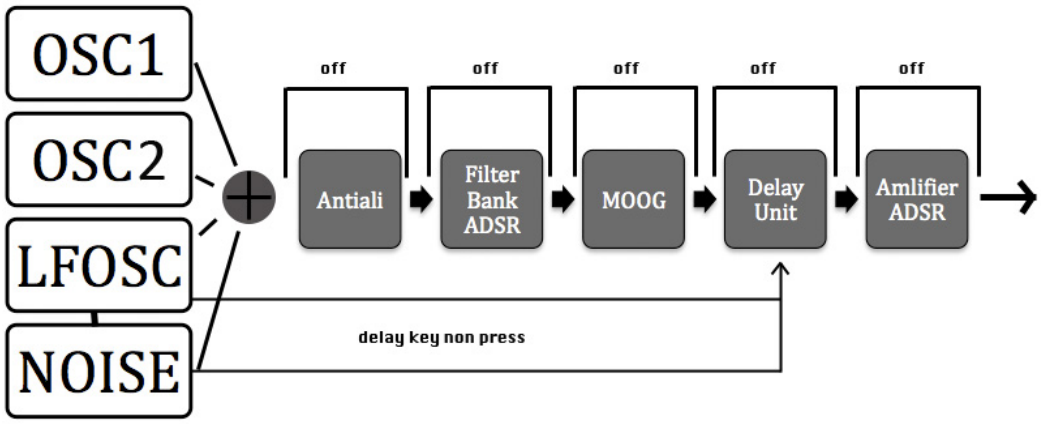


Figure 3-IV FulSynth complete block diagram

These are the main units:

OSC1: Bank of oscillators. Main signal source. Melodic purpose.

OSC2: Identical bank of oscillators. Main signal source. Melodic purpose.

LFOSC: Low Frequency Oscillator. Secondary signal source. Rhythm and background utility. Connected with the NOISE Unit.

NOISE: Random pink and white noise signal generator. Secondary signal source. Rhythm and background utility.

Antiali: Antialiasing filter. Graphically included in the Filter Bank interface. It can be bypassed.

Filter Bank: Low-pass, high-pass and band-pass filters controlled by an ADSR envelope generator (section 2.6.6). The unit can be bypassed.

MOOG: Pd emulation of the Moog synthesizer ladder filter. It may follow the existing signal chain or it can replace the Filter Bank rendering itself the only filtering unit in the chain. It can be bypassed.

Delay Unit: Introduces delay to its input signal. It can be bypassed.

Amplifier: ADSR controlled volume modifier stage before the main output.

It is of little value explaining with detail all code steps; there are attached printouts of all the functions commented into the appendix. However, the main blocks, tools, functionalities and features will be explained.

3.3. Oscillator

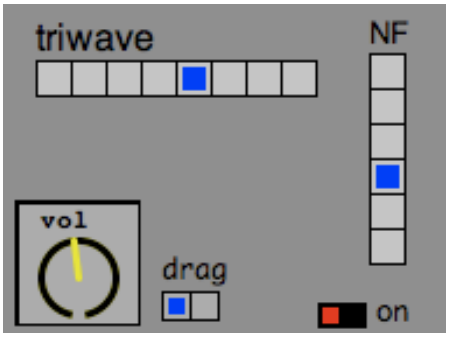


Figure 3-V FulSynth OSC1 and OSC2 abstraction

The oscillator banks are placed on the top-left corner of FulSynth's main interface (block 'A', section 3.1). They are two identical Pd independent abstractions sharing the same code (see *osc1_interface.pd into* appendix C). The objective of this unit is to generate the signal that will provide melody to the performer.

Eight different types of signals can be triggered. The keyboard selects the pitch and triggers either one or two added signals. The oscillator bank gives the possibility to choose among different octave ranges.

Although it is contrary to the subtractive idea of a synthesizer, there is a signal addition before the filtering stage. At the source the signals from the oscillators and the other two sources (LFO and/or NOISE) may be added. There is an addition before the harmonic modification takes place and gives the subtractive name to the method. At the source OSC1 and OSC2 are typically going to be combined to create a richer spectrum and more powerful initial signal.

One oscillator can create the following signals: *sine, sawtooth, inverted sawtooth, triangle saw, triangle, square wave, rectangular wave* and *pulse*. They can be among six different octave configurations and be set to a single volume.

Although there is no mathematical representation of them inside the current code, during the wave shaping development it was necessary to check with Pd that they had the right shape. The following are the demonstrations:

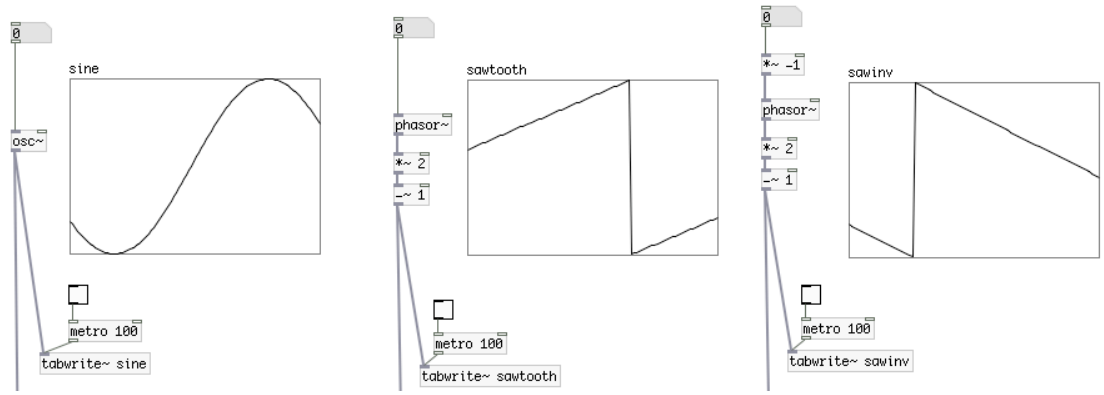


Figure 3-VI Waveshaping devlopement: sine, sawtooth, inverted sawtooth

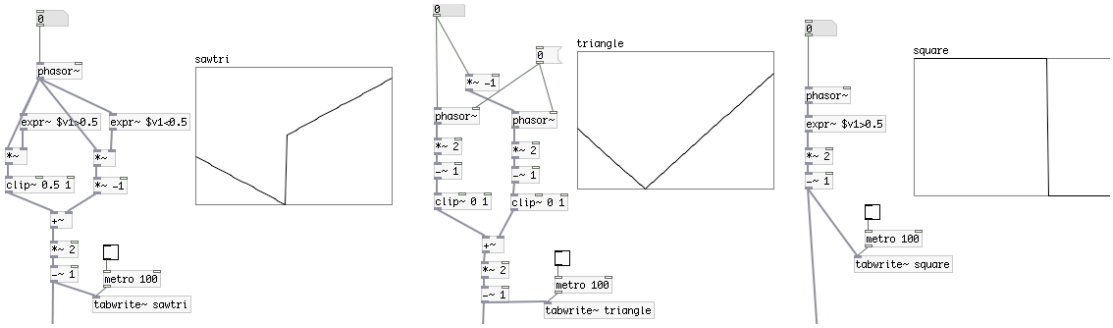


Figure 3-VII Waveshaping developement: trianle saw, triangle wave, square wave

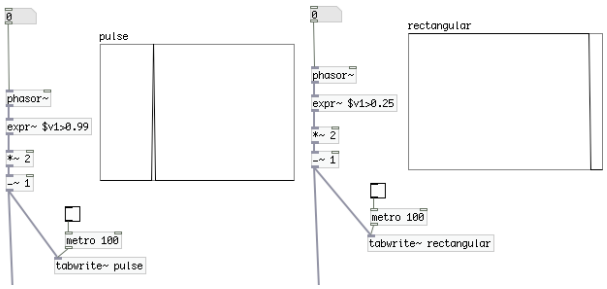


Figure 3-VIII Waveshaping developement: pulse, rectangular wave

Oscillators have a second function mode called *drag mode*. If it is operational each time a key is pressed the system will hold or drag the pitch. When the next key is pressed the drag-action repeats with the new pitch. This mode gives the opportunity of creating some more background sound. There will be sound even when the performer is not pressing any key. It gives more compositional resources.

After some inner-code communication or interference-cleaning considerations, the main signal source is sent to the antialiasing filter stage.

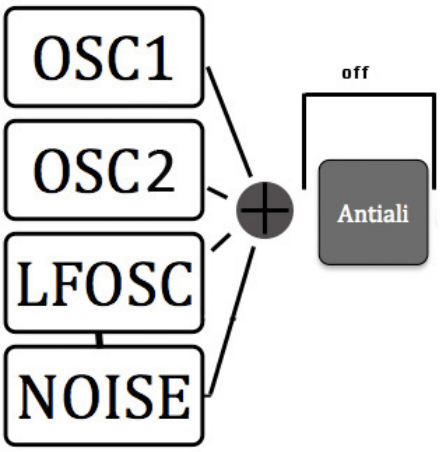


Figure 3-IX FulSynth block diagram: from the source to antialiasing filter

3.4. Antialiasing filter

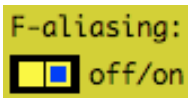


Figure 3-X FulSynth: *antialiasing* filter abstraction (see appendix)

The standard sample rate used to process audio signals is commonly 44100 samples per second, which is the speed at which CD are recorded. However, the programming considers that sampling speed might not be useful in all situations.

Human ear has an approximate audible range of 20Hz to 20KHz. Most of the signals Pd oscillators might mathematically produce have a wider harmonic content.

There are antialiasing filters placed at the computer soundcard. They affect the signal when it is still analogue, before it is discretized or digitally converted. To have a 44100-sampling rate system would avoid the aliasing interference produced only by the discretization of this incoming external audio signal. However, the signal is digitally generated from the inside. Working with Pd at the speed of 44100 samples per second might not be free from digital noise.

According to the Nyquist sampling theorem, the bandwidth of a signal must be less than half of the system's sampling rate in order to avoid aliasing (Wikipedia 2011) ('BW' and 'Fs/2', Figure 3-XI). A 44100-sampling rate gives a 22050-samples per second Nyquist frequency.

$$BW < Fs/2 \rightarrow Fs > 2*BW$$

Figure 3-XI Nyquist frequency

The sampling theorem (Figure 3-XI) defines the bandwidth *BW* and the sampling frequency *Fs* in an analogue context.

The following is an example in an analogue context: |X(F)| is the spectrum of an incoming audio signal x(t) arriving to the soundcard and fitting into the human audible range.

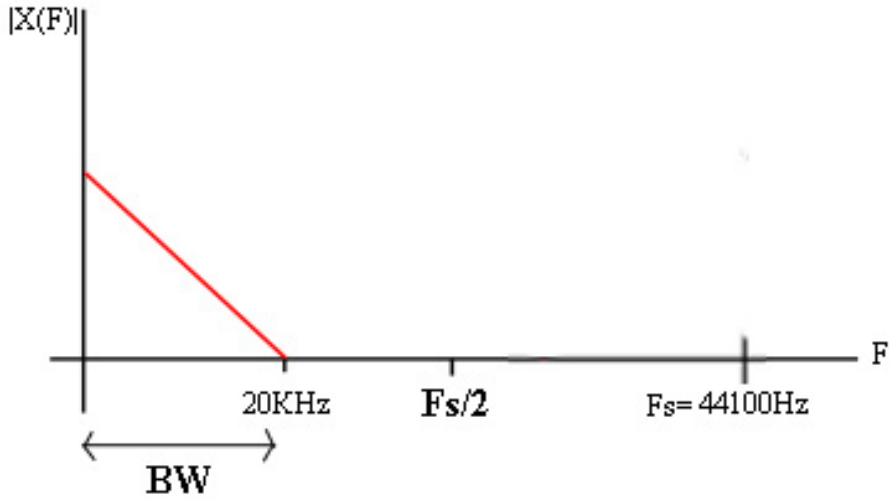


Figure 3-XII Nyquist graphic example

To obey the Nyquist conditions (Figure 3-XI) the soundcard's antialiasing filter is a low-pass filter with the cut-off frequency set at the half of the sample rate. The following is the ideal situation:

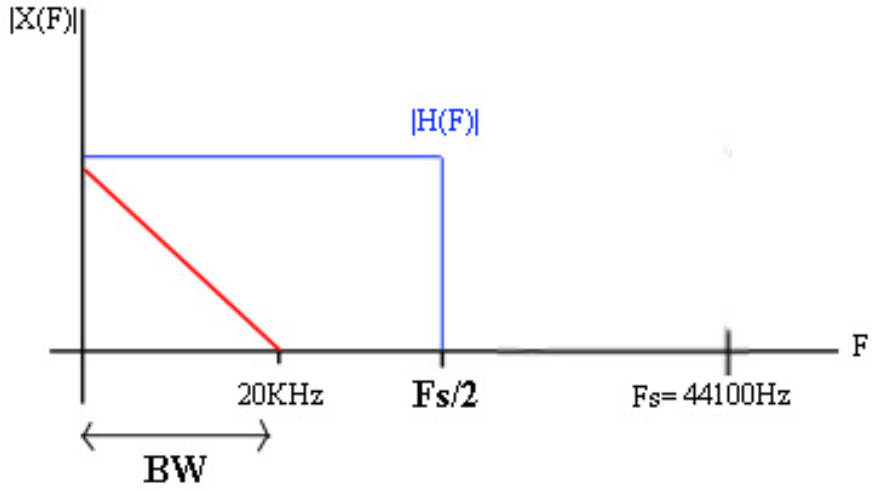


Figure 3-XIII Ideal soundcard's antialiasing filter

The discretization of an analogue signal implies the periodification of its spectrum. It is mathematically proven by applying the *Poisson Summation Fórmula* (Wikipedia 2011). After passing the antialiasing stage and becoming digitalized, the spectrum of the digitalized x(t) function is |X(f)|. Following the Nyquist-Shannon sampling theorem (Wikipedia 2011): *0.45*, *0,5* and *1* are the discrete values of the spectrum of the digital signal (Figure 3-XIV):

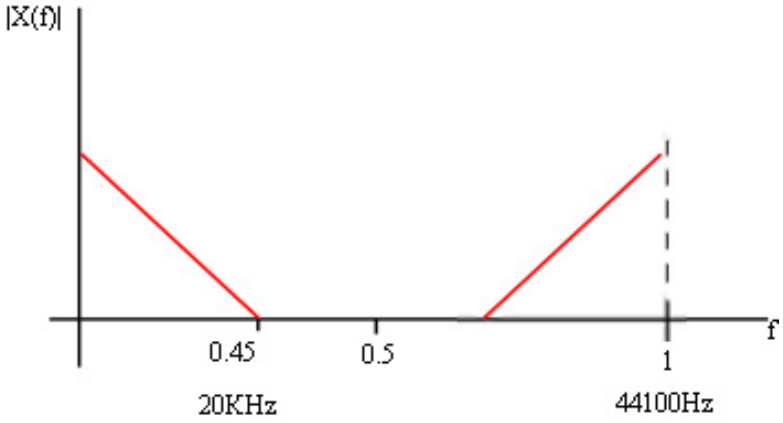


Figure 3-XIV Audio input signal digitalized

Antialiasing noise occurs when the periodifications of the spectrum intersect. Having a 44100-sample rate it is possible that Pd generates digital signals (in FulSynth oscillator bank: square wave, pulse wave, sawtooth, etc.) that spread its spectrum further than 0.45. One of these aforementioned signals sampled at 44100 samples per second could result in the following:

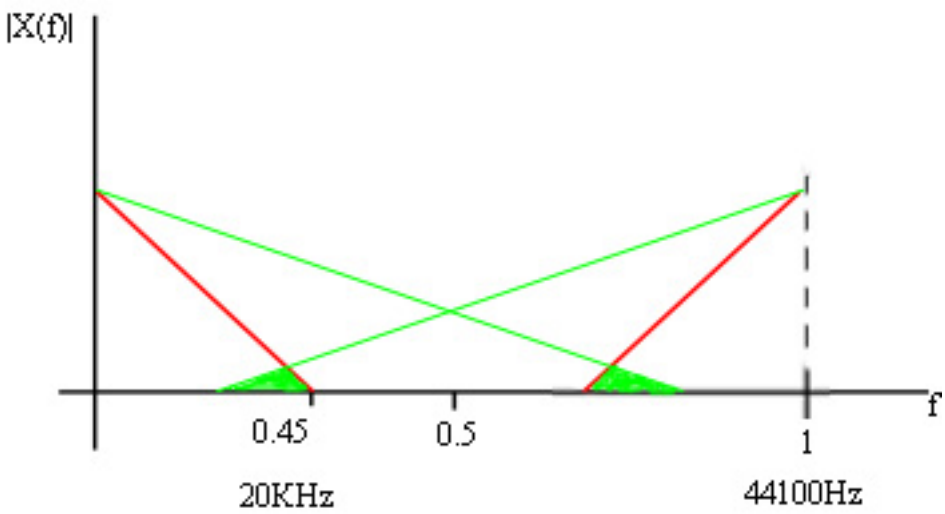


Figure 3-XV Aliasing

There is interference in the human audible range. There is a “folding over”. It is possible to notice an increase of digital noise especially in high frequency signals with a rich spectrum bypassing FulSynth’s antialiasing filter, and this is heard more easily with headphones.

The solution to the aliasing problem can be solved through *oversampling* (Puckette 1996) and low-pass filtering again. Pure Data may change a subpatch sample rate thanks to *block~* or *switch~* components. This means that the signal going into that subpatch is sampled at another speed and when the signal leaves the subpatch is resampled at the initial speed to keep on working with the rest of the program.

In that context and according to Figure 3-XV, oversampling 16 times a 44100-sample rate gives an analogue frequency of 44100*16 = **705600**Hz: a new sampling rate. The 0.45 digital value is now also in analogue context 20KHz*16= **320KHz.** Then, applying the Nyquist formula and a a low-pass antialiasing filter with 20KHz cut-off frequency, results in the following:

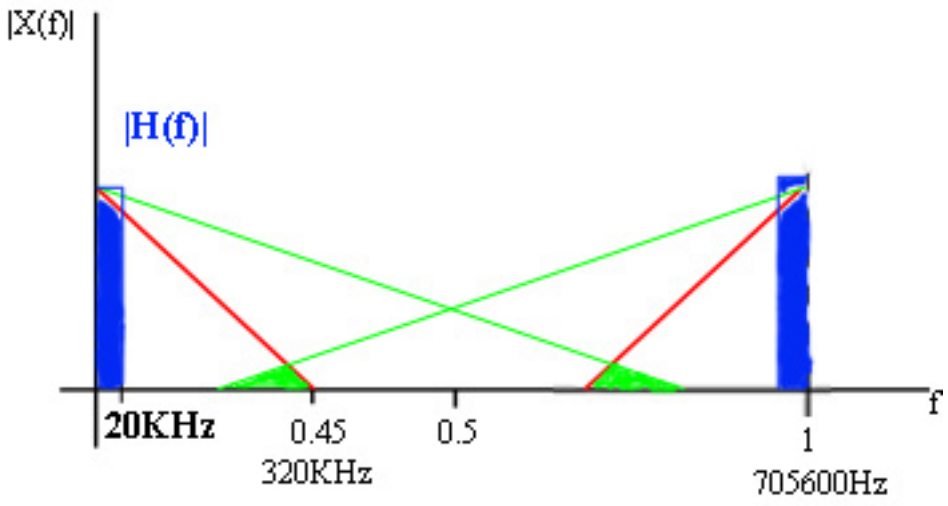


Figure 3-XVI FulSynth antialiasing filter, ideal situation

The self-designed antialiasing filter function oversamples the 44100-rate per 16 and applies a Butterworth low-pass antialiasing filter to erase the worthless frequency content above 20KHz. This frequency range is worthless because it cannot be heard. It would fold over the human hearing bandwidth creating aliasing noise. That “folding over” is due to the periodic spectrum a discrete signal has.

After filtering, the remaining signal is processed back to 44100-sample rate. The interference has been attenuated. The signal leaves the oversampled subpatch and comes back to the main program before being addressed to the ADSR filter bank.

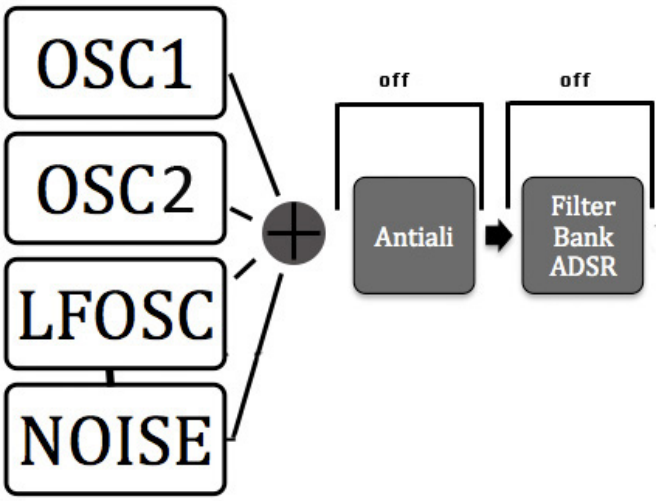


Figure 3-XVII FulSynth block diagram: from the source to ADSR Filter Bank

3.5. ADSR Filter Bank

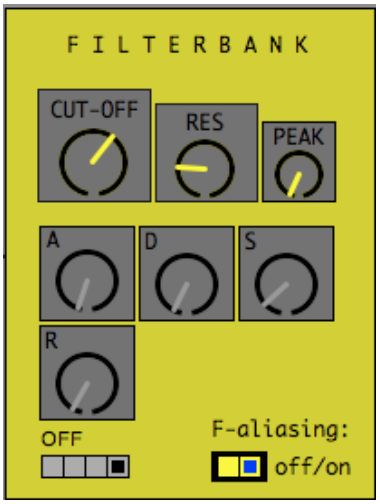


Figure 3-XVIII FulSynth: ADSR filter bank abstraction

Here, the four different sound sources, OSC1, OSC2, NOISE and LFO can be harmonically modified. The ADSR filter bank performs the subtractive action through three filtering possibilities. Each one includes an ADSR envelope generator, which modulates the cut-off frequency as explained previously (section 2.6.6).

The Filter bank includes a low-pass, high-pass and band-pass filter. It is possible to control the *cut-off* frequency ('CUT-OFF', Figure 3-XVIII) of each filter and its *resonance* ('RES', Figure 3-XVIII) to get different timbre modifications. It is possible to change the *attack time* ('A', Figure 3-XVIII), to set the highest *peak* the signal gets after the attack time ('PEAK'), the *decay time* ('D') the *sustain time* ('S') and the *release time* ('R').

Although filters will act on any input signal, the ADSR envelope generator is linked to a triggering event (see *pd adsr* subpatch into appendix). FulSynth is designed using a midi or computer key-event as a trigger (see *keypress.pd, midikey.pd* into appendix). Due to that feature, although the NOISE Generator and the LFO will be always affected by the filter stage, they are not going to be modified by the ADSR envelope generator unless a key is pressed.

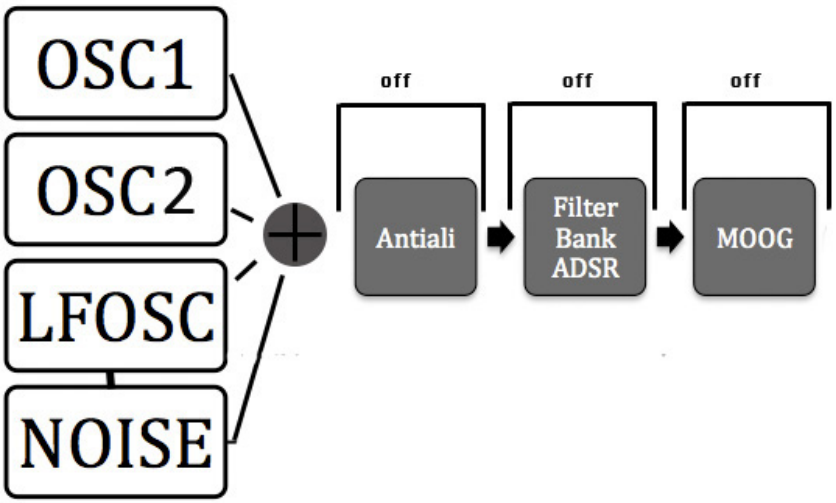


Figure 3-XIX FulSynth block diagram: from the source to MOOG Unit

3.6. MOOG Filter Unit

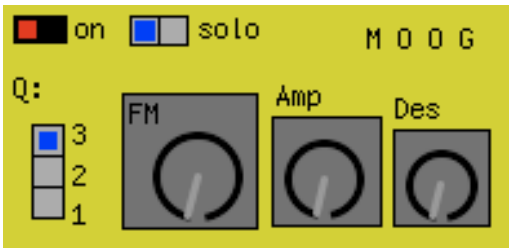


Figure 3-XX FulSynth: MOOG Filter Unit abstraction

FulSynth MOOG low-pass Filter Unit works into two possible modes: taking the outcoming filter bank signal, or avoiding the whole ADSR filter bank, becoming the only filter stage in the synthesizer (see the subpatches *moog.alone, moog.addition* into the appendix).

FulSynth MOOG Filter Unit includes three control parameters: 'FM', 'Amp' and 'Des' to modulate its cut-off frequency (Figure 3-XX). The 'Q' switch sets the filter resonance, '3' being its sharpest value.

Example (Matlab 2008):

1. A sine (f= FM) wave is emulated

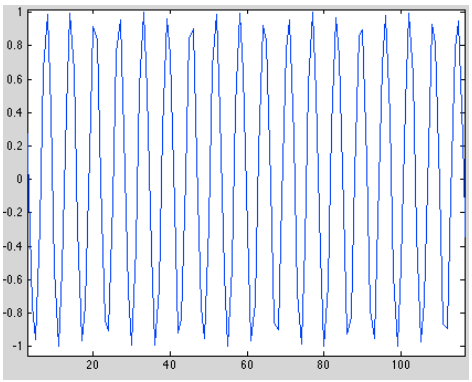


Figure 3-XXI Initial FM frequency sine signal

2. Add 1 to get positive values

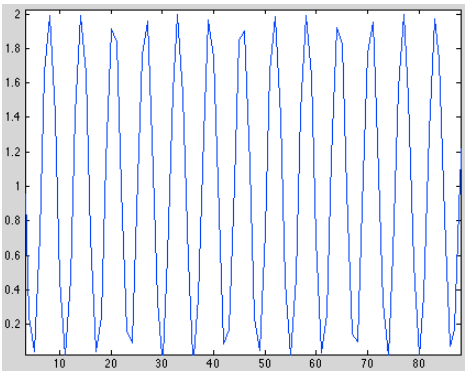


Figure 3-XXII Initial sine in positive values

3. The amplitude is set up

4. 'Des' fixes a deviation centre point

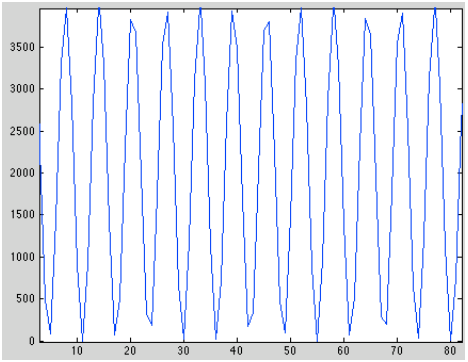


Figure 3-XXIII Setting AMP=2000

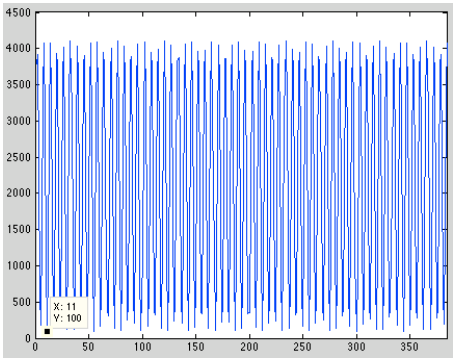


Figure 3-XXIV Setting DES=100 (offset)

The MOOG Unit cut-off frequency is centred at fc= 2100Hz (DES=100) and it oscillates at FM=1Hz, within a maximum deviation from its central point of AMP=2000. The FulSynth MOOG low-pass filter unit has therefore an "automatic" variable cut-off frequency. In this example it goes from 100Hz to 4100Hz and the cycle is repeated every second (see Figure 3-XXVI).

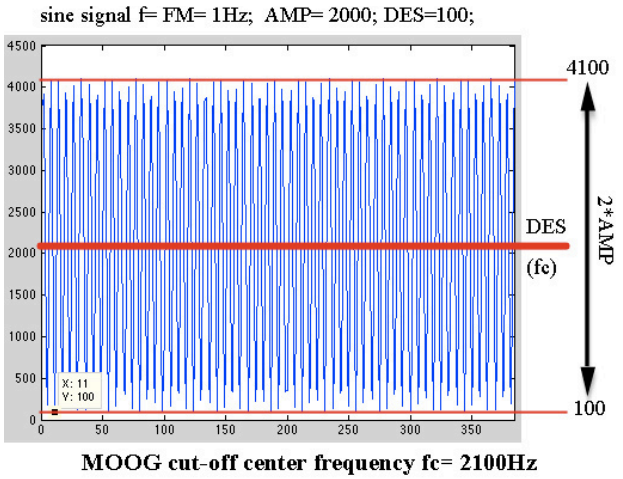


Figure 3-XXV MOOG unit filter fc route

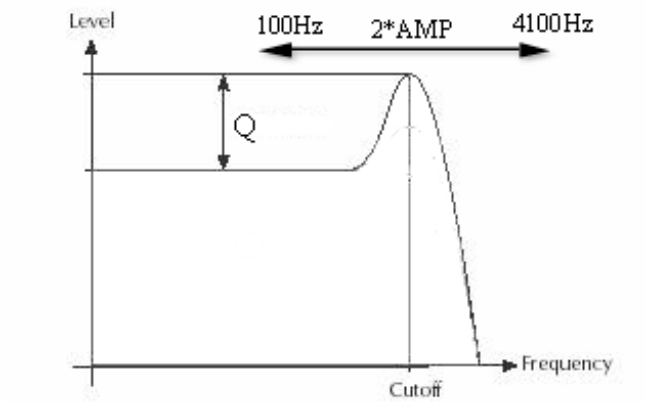


Figure 3-XXVI MOOG unit frequency response (Samplecraze 2010)

This is obviously just a technical explanation for a process the performer will decide to adjust only taking in consideration his musical or hearing requirements. However, this example illustrates one important concept a synthesizer performer must comprehend.

3.6.1. Modulation

The online dictionary Your Dictionary (LoveToKnow 2011) defines *modulation* as:

> *a variation in the amplitude, frequency, or phase of a wave in accordance with some signal*

FulSynth low-pass filter MOOG Unit has hence a modulated cut-off frequency. A low 'FM' frequency oscillator (Figure 3-XX) sets the "automatic" cut-off frequency. Subtractive synthesizers use signals to control other signals parameters (modulation) in many situations. This might be confusing, but it is often the essence of techniques and sound expressivities.

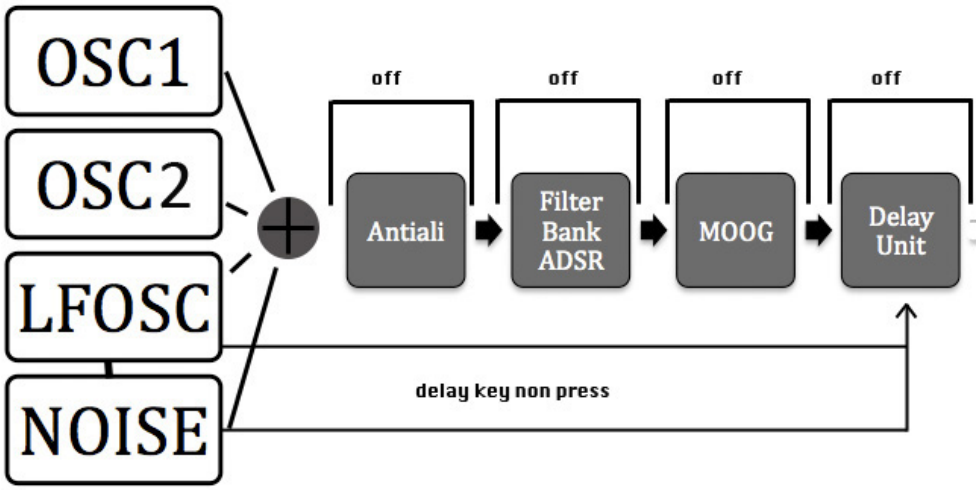


Figure 3-XXVII FulSynth block diagram: from the source to Delay Unit

3.7. Delay unit

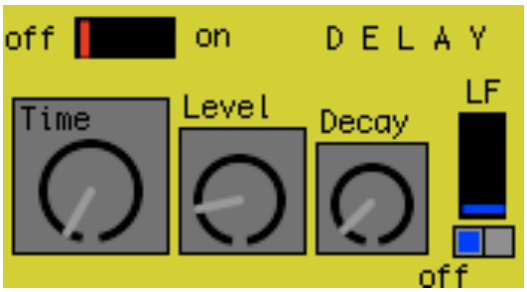


Figure 3-XXVIII FulSynth: Delay unit abstraction

The signal arrives into the Delay Unit from two different sources: *sig.in* comes from the oscillator-modifier chain when a key is pressed; noise and/or LFO signal arrive by *sig.non.press*, and have its own delay chain not constrained to any key action (see *delay-unit.pd* into appendix).

The 'Time' parameter (Figure 3-XXVIII) controls the time the repetition of the signal will take to be produced. The 'Level' parameter sets the gain or volume of the signal delayed, its maximum level being equal to the signal's non-delayed level. When the 'Level' is at the maximum value there is no attenuation for the delayed signal. The delay therefore becomes a loop.

The 'Decay' time parameter acts through an ADSR envelope generator controlling amplitude; a short decay time produces a more percussive sound.

Instead of using switches, sliders give the option of introducing the Delay Unit into the main chain of events progressively. The red switch is for the key-dependent actions and the blue one for the LFO and NOISE Generator sources.

The Delay Unit is useful to fulfil backgrounds with echo and create rhythms or just to create different lines of melody. It gives one more sound possibility to a synthesizer.

3.8. NOISE Generator Unit

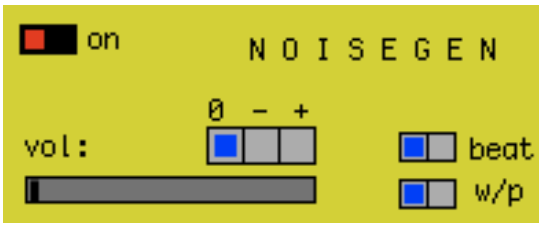


Figure 3-XXIX FulSynth: NOISE Generator Unit abstraction

Subtractive analogue synthesizers often include a random signal source with a volume control. Pure Data can generate both random white and pink noise. This feature has been implemented in FulSynth and it is controlled from the NOISE Generator Unit (Figure 3-XXIX).

The NOISE Generator Unit includes also a rudimentary, experimental and automatic beat machine that follows one Low Frequency Oscillator Unit beat envelope (see *noiseryhthm.pd* into appendix). The blue switch "beat" activates this feature.

In the centre of the interface there are three switches. These are quick-access volume controls.

Combining NOISE Generator Unit with the rest of the synthesizer features it is possible to generate richer backgrounds, new sounds and rhythms. It is important to remark that the possibilities of any instrument are fixed by its physical characteristics, but performer skills will decide at last its entire capacity of expression.

3.9. Low Frequency Oscillator (LFO)

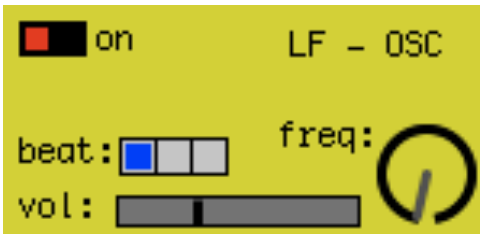


Figure 3-XXX FulSynth: Low Frequency Oscillator abstraction

This is a simplification of the main oscillator source and, although it is able to work as a third oscillator, it has been designed to accomplish mostly a rhythmical

function. The core of the LFO unit is in the abstraction *oscbank-LF.pd* (see appendix). It generates a square wave, a pulse train and a rectangular wave and these are the three different kinds of beats and beat-switch positions (Figure 3-XXX). The unit creates beats when the frequency control is set down to a low level. A low-pass filter processes the beats either to emphasize the common low frequency beat characteristics or to allow a brighter solution.

It is possible to choose among three different kinds of beats and to control their frequency and volume. To switch on the engine of the already mentioned experimental beat-noise machine (section 3.8) it is necessary to choose the first beat, on the left at the three-positions switch. The LFO volume must not be too high. One *snapshot~* command is working here as an envelope detector and sending bangs to the NOISE Generator Unit (see *osc-LF.pd* into appendix). If the first beat is chosen and the volume is right it will creates a simple 4/4 beat rhythm by moving the Noise Generator Unit volume value up and down (to zero) (the 'beat' switch must be activated in the NOISE Generator Unit, Figure 3-XXIX).

The last part to introduce is the ADSR Amplifier stage, completing finally FulSynth block diagram:

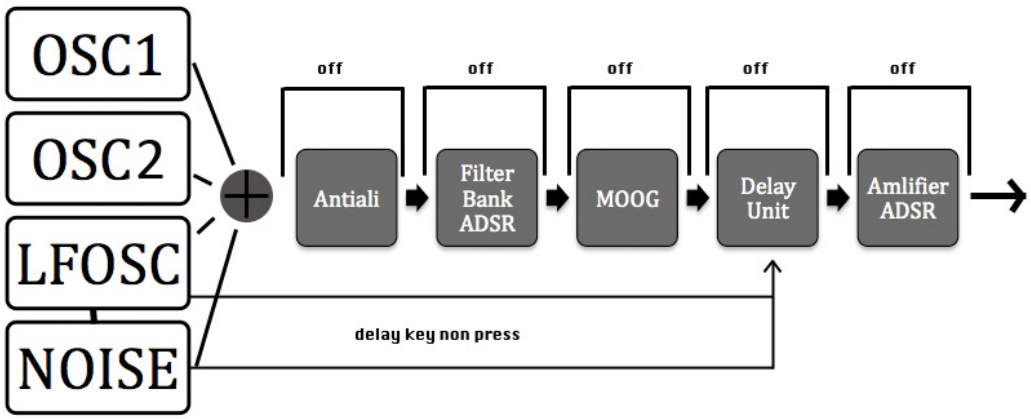


Figure 3-XXXI FulSynth complete block diagram

3.10. Amplifier ADSR

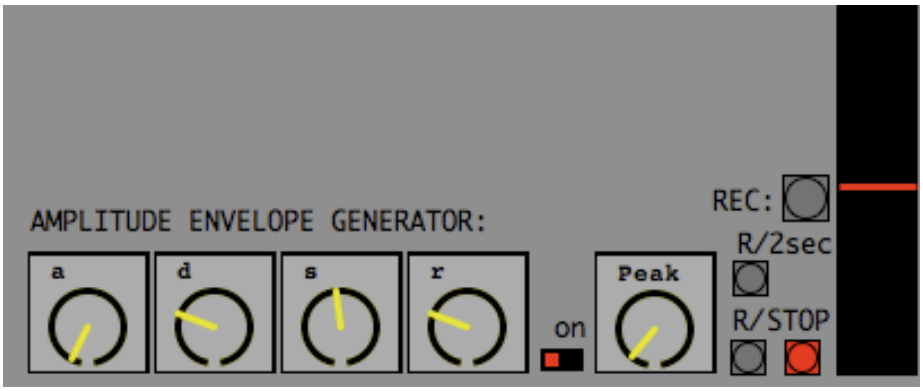


Figure 3-XXXII FulSynth: Amplifier and main output abstraction

At the final stage, after passing through Delay Unit, there is the ADSR Amplifier Unit. It is the second modifier commonly found in a basic subtractive synthesizer chain.

Keeping in mind the subtractive synthesis source-modifier theory of manipulating sound:

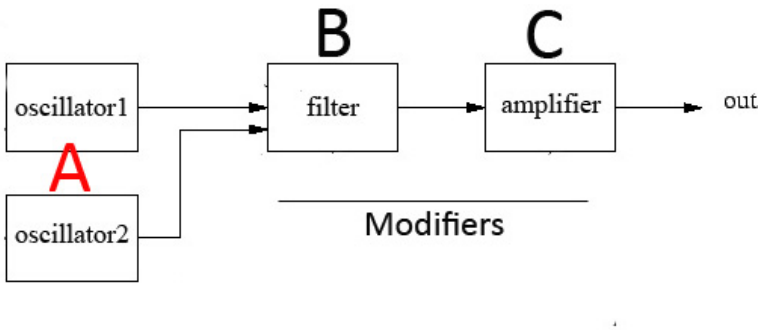


Figure 3-XXXIII Source-modifiers. Subtractive synthesis block diagram

The ADSR Amplifier Unit controls the amplitude of the final signal (section 2.6.6). A master volume control is combined with the ADSR envelope generator, which can be switched on or off, to set the signal final gain (Figure 3-XXXII). To avoid overdrive, sound signal in Pd must be limited from -1 to 1.

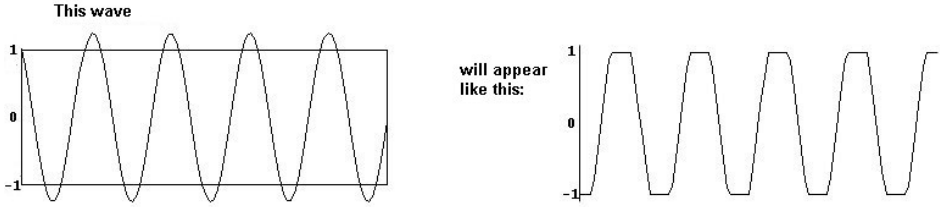


Figure 3-XXXIV Overdrive (Kreidler 2009)

The signal is thus limited and is also high-passed through a 1Hz cut-off filter, which removes the unwanted offset. After this, the signal is finally sent to the *dac~* object, which will convert (digitally to analogue) the negative values -1 to 0 to silence and the values from 0 to 1 range to sound, using the speakers and the computer soundcard (Kreidler 2009).

In order to realize tests and comparisons in the next chapter, it has been useful to include a recorder (REC) option (Figure 3-XXXII). 'REC' opens a save file panel. It is possible to record a 2 seconds sample or to start and stop one chosen recording length.

3.11. Keyboard control

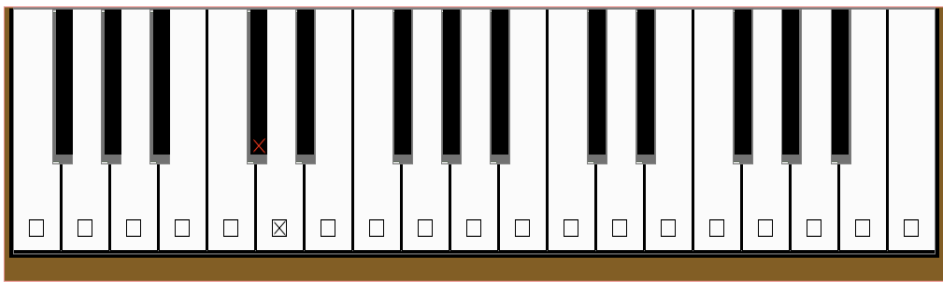


Figure 3-XXXV FulSynth: keyboard abstraction

A graphic piano keyboard representation gives the software a good appearance and shows the user the note that is being played. The Keyboard control can be driven either from a midi keyboard or from a computer keyboard (see *midikey.pd, keypress.pd* and *keys_receive.pd* functions into the appendix.).

Its functionality has been developed in reference to a QWERTY computer keyboard, being the keys from 'z' to 'k' separated one octave; 'z' corresponds to C2 (65.4Hz) frequency, while 'k' to C3 (130.8Hz), if it is chosen the initial and normal frequency mode (NF) in the oscillator interface (section 3.3). This is the only range in which keys will be light up on the screen:

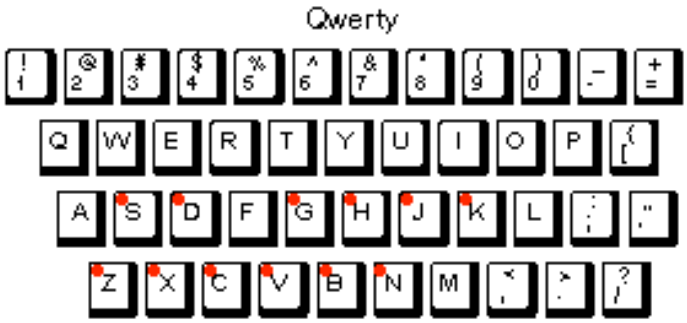


Figure 3-XXXVI Active keyboard keys (En el tintero de Zamzara 2006 Blog)

Sending different frequencies, for instance playing a midi keyboard in a higher octave, would produce sound, but will not affect the graphic interface (Figure 3-XXXV).

Although the main keyboard-related functions have already been mentioned in this section, the whole keyboard is full of abstractions of different canvas layers, which allow the inner communication and improve the final graphic representation.

CHAPTER 4. TESTING AND COMPARISONS

4.1. Reference points

The following synthesizers are going to be compared with FulSynth. They have already been quickly introduced as well-known software virtual subtractive synthesizers (section 2.5). Emulated by prestigious software developer labels, they have been chosen among many other possible interesting devices. They all present a subtractive synthesizer source-modifier block diagram and they will set reference points to compare, understand, discuss and learn from.

FulSynth is going to be compared with:

1. **Novation Bass Station** 1.5 (Novation 2007)
2. **Arturia Minimoog** 1.6 (2006) (Arturia 2009)
3. **AudioRealism Bass Line Pro** 1.102 AU Intel (ABL Pro) (Realism 2009)

Due to their great functionalities, it is not possible here to completely analyse them. However, testing them will be useful to highlight their qualities and differences with FulSynth, especially in relation to their common features.

4.2. TEST1 - Waveshaping

4.2.1. Oscillator Blocks

Figure 4-I Oscillator blocks in: Bass Station (top left) – ABL Pro (bottom left) – Minimoog (right)

A new user gets rapidly familiarized with the oscillator part of any device because from one synthesizer to another there is no much difference; they all share the typical parameters.

It is common to use knobs or switches to set the wave shape of the signal source, its frequency and other features that might be related to it. It is always essential to read the labels in order to recognize the right knobs.

Figure 4-II Oscillators for each synthesizer - Bass Station (top) - Minimoog (middle) - ABL Pro (bottom)

4.2.2. Waveshaping

In order to analyze a plain signal produced with a synthesizer, the filtering stage should be deactivated. However, it is not possible to deactivate the filter in the analysed devices, only in FulSynth. Each filter has been opened at its maximum cut-off frequency.

Each device produced three waves, three different frequency square waves: one low frequency, one mid frequency and one high frequency wave. They have been recorded with generic audio software and analyzed with AudioXplorer (Arizona Software 2006).

1. **Novation Bass Station** 1.5 (Novation 2007)

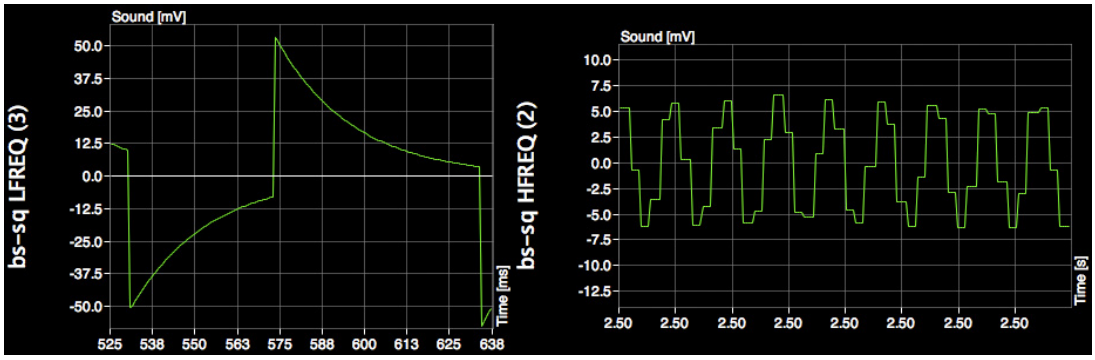


Figure 4-III Novation Bass Station LOW (left) and HIGH (right) frequency square wave

LOW: T= 113ms ; f_L**= 8.8Hz** ; 16”

HIGH: T= 1.11ms ; f_H**= 900Hz** ; 2”

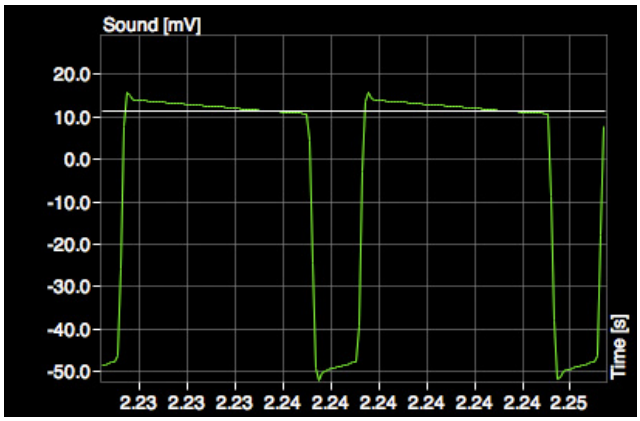


Figure 4-IV Novation Bass Station MID frequency square wave

MID: T= 8.25ms ; f_M**= 121.2Hz** (detuned C3) ; 16” ; keyboard oct. decrease

2. **Arturia Minimoog** 1.6 (2006) (Arturia 2009)

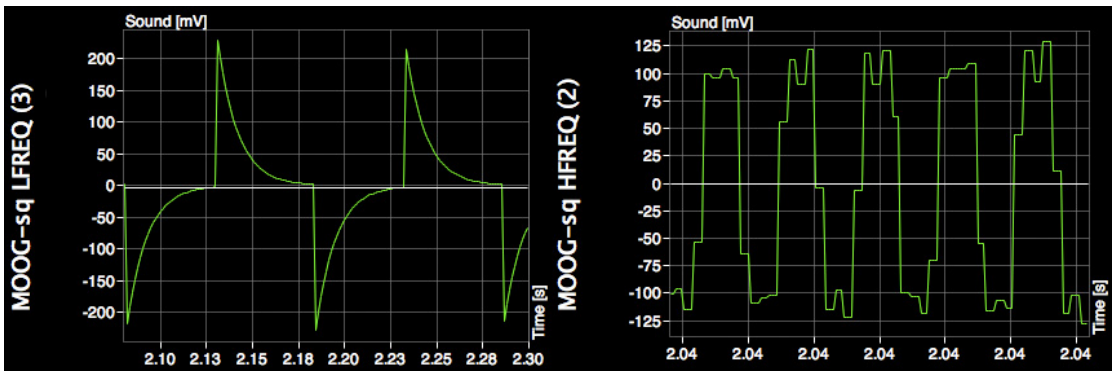


Figure 4-V Arturia Minimoog LOW (left) and HIGH (right) frequency square wave

LOW: T= 105ms ; f_L= **9.5Hz** ; 32” ; detuned

HIGH: T= 0.2ms ; f_H= **5kHz** ; 2”

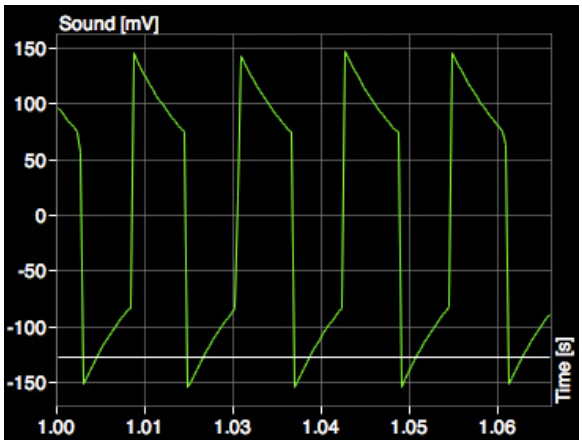


Figure 4-VI Arturia Minimoog MID frequency square wave

MID: T= 15,3ms ; f_M= **65.3Hz** (C2) ; 16”

3. **AudioRealism Bass Line Pro** 1.102 AU Intel (ABL Pro) (Realism 2009)

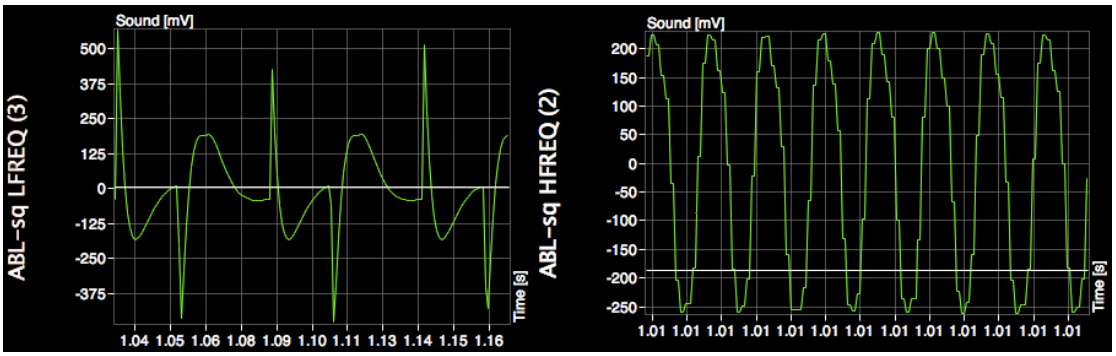


Figure 4-VII ABL Pro LOW (left) and HIGH (right) frequency square wave

LOW: T= 54.6ms ; f_L= **18Hz** ; 16” ; detuned

HIGH: T= 0.25ms ; f_H= **4kHz** ; 4” ; detuned

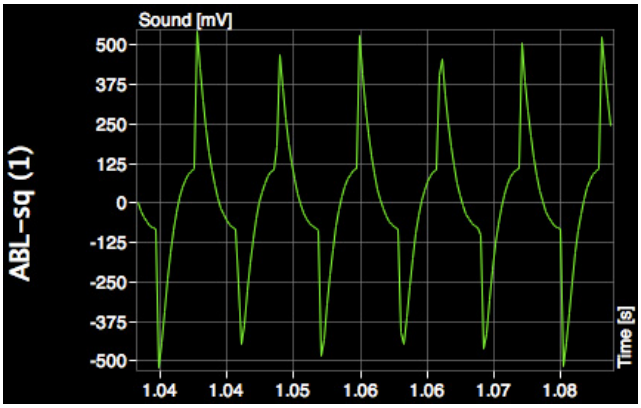


Figure 4-VIII ABL Pro MID frequency square wave

MID: T= 7.8ms ; f_M= **128.2Hz** (C3 slightly detuned) ; 16”

4. FulSynth

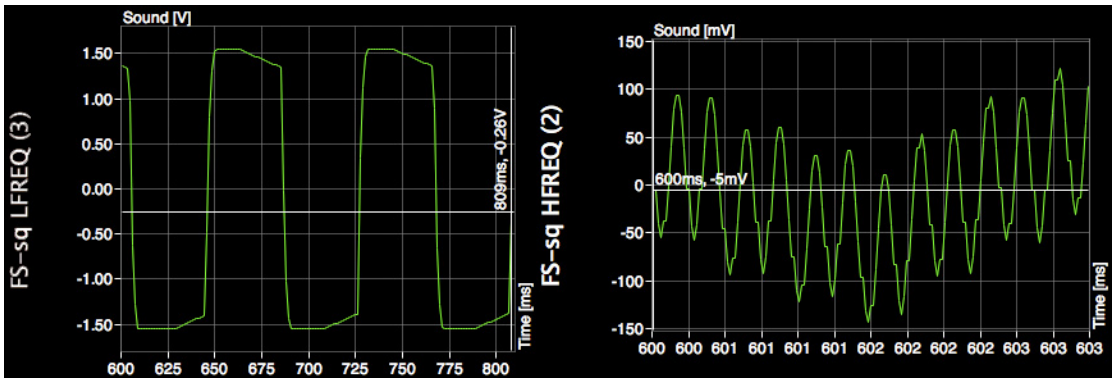


Figure 4-IX FulSynth LOW (left) and HIGH (right) frequency square wave

LOW: T= 84ms ; f_L= **11.9Hz** ; ‘:2’

HIGH: T= 3ms ; f_H= **4kHz** ; ‘x8’

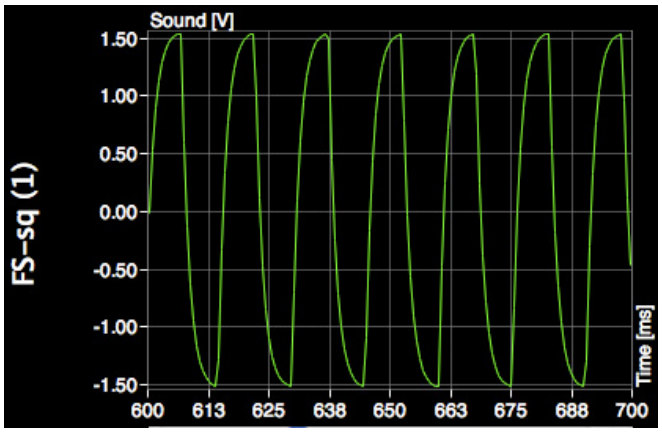


Figure 4-X MID frequency square wave

MID: T= 15ms ; f_M= **66.6Hz** (C2)

4.2.3. Preliminary conclusions

The waves would be analyzed with more precision and detail using a fully equipped electronics laboratory. However, AudioXplorer (Arizona Software 2006) allows a plausible first reading.

LOW frequency square wave:

Here, FulSynth shapes a square wave. Novation and Arturia use the same kind of approximation: two abrupt transitions combined with an exponential decrease set the cycle. AudioRealism uses two abrupt transitions to characterize the waves but combined this with an exponential and parabolic function.

HIGH frequency square wave:

All four emulations show at high frequencies the effect of bit quantification. FulSynth's high frequency square wave presents a low frequency interference modulating the square wave. Arturia Minimoog emulates a closer-shaped square wave, followed in ranking first by Novation Bass Station and subsequently by AudioRealism.

MID frequency square wave:

FulSynth approximates the square wave through exponential transitions. Novation creates a clear square but some peaks appear on its edges. Arturia seems to expand somehow its low-frequency criteria to develop its personal clean-shaped approximation of square wave, as it does AudioRealism, keeping the exponential-parabolic unclear shape.

4.2.4. Comparisons: Novation Bass Station

Figure 4-XI Novation Bass Station Oscillators

Depending on the goals or the style of the synthesizer, an oscillator block may differ a little bit. Novation for instance decided to create the Bass Station synthesizer using only two main signals: a sawtooth and a square wave (pulse-width modulated) (Figure 4-XII). These previously explained signals (section 2.6) have the richest harmonic content amongst those one might find in a synthesizer. Bass Station bases the source of its sound on the useful and most distinctive waves of subtractive synthesis. However, at first sight the Bass Station oscillator bank might be unclear. Doubts quickly disappear after testing it, using experience of other similar devices, or reading its instrument's user guide manual. Bass Station manual is remarkably helpful to beginner subtractive synthesizer users and offers the instrument answers any user might eventually need (Novation 2007).

Comment: The octave range in all three commercial synthesizers is referred in 'feet', a convention which comes from the length of organ pipes. FulSynth octave-range labels are defined by a logical mathematical convention. FulSynth initial or normal frequency (NF, Figure 4-XIII), has been fixed taking the reference of a C2 note in the lowest C of the FulSynth keyboard. FulSynth is thus to be born as a bass-pitched instrument.

Increasing the octave range in one octave from 'NF' the label would show 'x2', decreasing one ':2', etc. It might be less confusing for a beginner performer (see Figure 4-XIII).

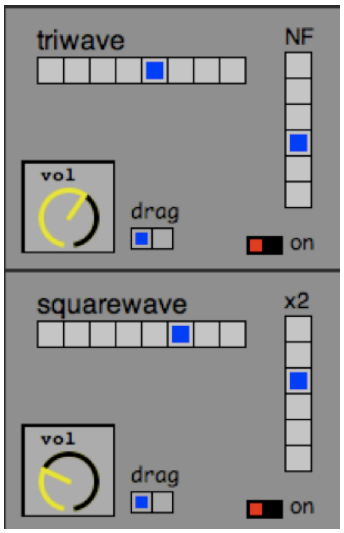


Figure 4-XII FulSynth oscillators

FulSynth clearly distinguishes between two oscillators. Besides, Bass Station uses a 'MIX' control that sets the balance between the two oscillators.

Many synthesizers use a mixing stage. Instead of controlling the mix through oscillator one and two knob-volumes (Figure 4-XIII) only one knob here achieves the same function.

Bass Station fixes its first oscillator with two possible wave shapes to an 8" octave range (Novation 2007). This is the "base" oscillator. The second oscillator is more flexible and it could be thought of the "changing" one. It can be set among four different octave ranges, it can be *detuned* from one selected octave range (alterations from the keyboard pressed note of frequency cents) or it can be altered by *semitone*, which would provide the option of stronger frequency alterations, making it possible to create pleasant musical intervals, a perfect 4^{th}, 3^{rd} minor, etc. (Figure 4-XII).

FulSynth does not incorporate any modulation tool in its oscillators bank. Modulation is strongly used in many different ways in a subtractive synthesizer, and so it is in Bass Station:

Figure 4-XIII Bass Station oscillator modulation controls

The 'ENV' knob sets the amount of pitch sweep from the ENVELOPE 2 Bass Station ADSR envelope generator.

The 'LFO' knob sets the amount of pitch sweep by the Bass Station low-frequency oscillator. Therefore, one LFO modulates an oscillator frequency.

The ‘PULSE W’ knob controls the width of the Bass Station square wave, and it has three possible procedures. ‘MAN’: the knob will manually control the pulse width; ‘LFO’ or ‘ENV 2’: if either of them are switched the width of the square wave will be modulated by a special pulse modulation LFO or by the ADSR ENVELOPE 2.

4.3. TEST2 - Filtering

4.3.1. Filter Blocks

Figure 4-XIV Filter Blocks in Bass Station (top left) – Minimoog (top right) – ABL Pro (bottom)

The easiest way to find the filter bank in any device is to first look for a ‘filter’ printed label. However, as it happens in ABL Pro, there is no such a label (Figure 4-XV). The best reference point to find the filter and its related controls is then the cut-off frequency knob. Cutting-off the keyboard from all three devices:

Figure 4-XV Situation of the filter for each synthesizer- Bass Station (top)- Minimoog (middle)- ABL Pro (bottom)

4.3.2. Filtering

The filtering features of each device have been tested with AudioXplorer (Arizona Software 2006). The experiment consists of filtering a square wave twice: first with one completely closed filter (fc_{LOW}), and secondly with one completely open (fc_{HIGH}). The first measure is done with a low frequency square wave, the second one uses a mid-high frequency wave. The resonance parameter is always set to its minimum value. The FulSynth filter slope is about 12dB/oct (see Appendix A) while the other three devices share the slope value of 24dB/oct.

1. **Novation Bass Station** 1.5 (Novation 2007)

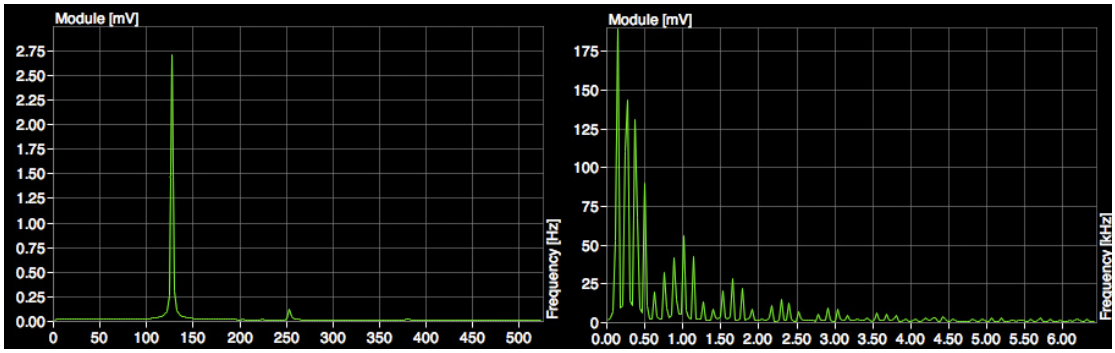


Figure 4-XVI Novation Bass Station LOW (left) and HIGH (right) filter cut-off frequency

fc_{LOW} - spectrum output max amplitude: 2.75mV

fc_{HIGH} - frequency spectrum width: 7kHz

2. **Arturia Minimoog** 1.6 (2006) (Arturia 2009)

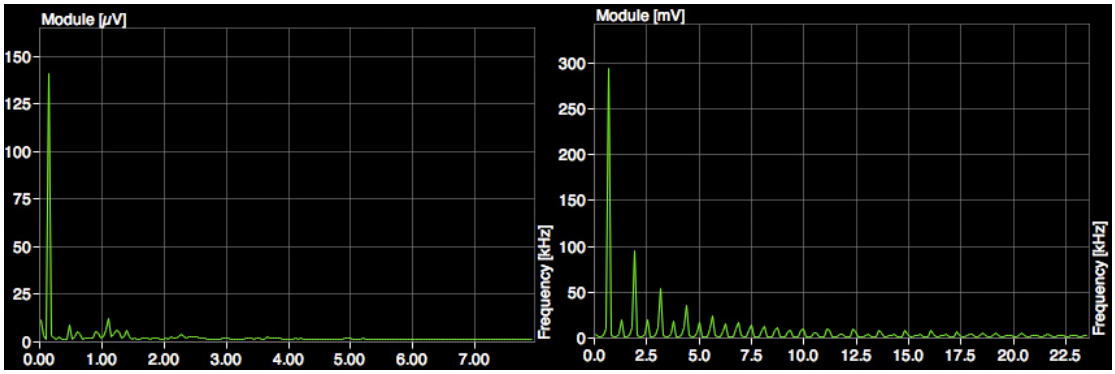


Figure 4-XVII Arturia Minimoog LOW (left) and HIGH (right) filter cut-off frequency

fc_{LOW} - spectrum output max amplitude: 140µV (approx.)

fc_{HIGH} - frequency spectrum width: 22.5kHz

3. **AudioRealism Bass Line Pro** 1.102 AU Intel (ABL Pro) (Realism 2009)

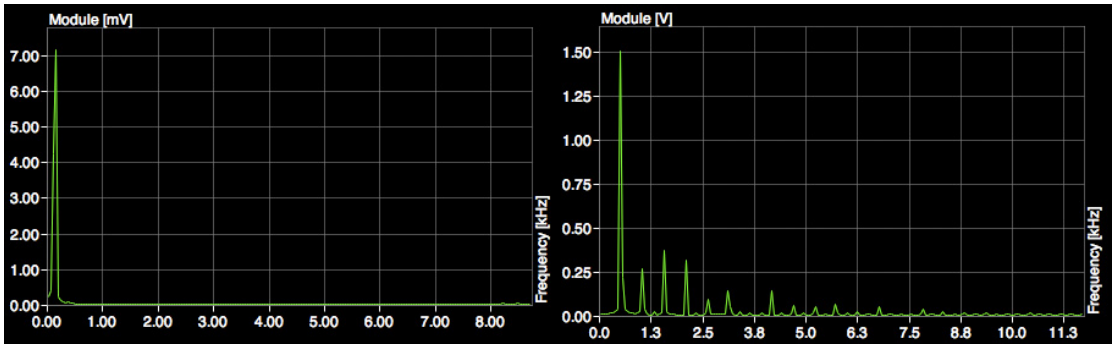


Figure 4-XVIII ABL Pro LOW (left) and HIGH (right) filter cut-off frequency

fc_{LOW} - spectrum output max amplitude: 7mV (approx.)

fc_{HIGH} - frequency spectrum width: 11.3kHz

4. FulSynth

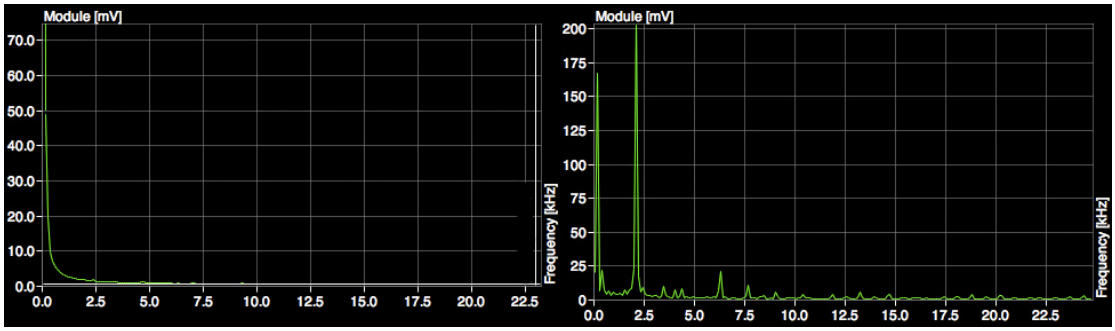


Figure 4-XIX FulSynth LOW (left) and HIGH (right) filter cut-off frequency

fc_{LOW} - spectrum output max amplitude: 70mV (approx.)

fc_{HIGH} - frequency spectrum width: 22.5kHz

4.3.3. Preliminary conclusions

The waves would be analyzed with more precision and detail using a fully equipped electronics laboratory. However, AudioXplorer (Arizona Software 2006) makes possible a plausible first reading.

Closed filter (fc_{LOW}):

FulSynth presents a 70mV output amplitude when the filter is supposed to cut-off everything. ABL Pro allows 7mV to pass, Novation Bass Station 2.75mV and Arturia Minimoog emulation offers a 140µV output amplitude, certainly the best at cutting-off.

The Arturia Minimoog closed filter differs by power of 10 from its value when the filter is open: 140µV are negligible and can be approximated to 0.

Open filter (fc_{HIGH}):

FulSynth presents an unwanted low frequency interference with an important amplitude level. Although its spectrum spreads until 22.5kHz, its harmonic content is not defining clearly a square wave spectrum.

Novation Bass Station spectrum spreads until 7kHz. The theoretical spectrum shape of a square wave is present in its harmonic content; however, Arturia Minimoog defines it much more clearly. Minimoog spreads its spectrum until 22.5kHz, the audible range, and presents a well-balanced harmonic content. The amplitude of the partials seems to smoothly decrease from the fundamental level. ABL Pro shapes a clean spectrum until 11.3kHz, but its harmonic content does not achieve Arturia Minimoog's balance.

4.3.4. Comparisons: Arturia Minimoog

Figure 4-XX Arturia Minimoog filter

Arturia emulates the 24dB/oct Minimoog low-pass filter. The cut-off frequency knob is next to the 'FILTER EMPHASIS' or resonance (Q) knob. 'AMONT OF CONTOUR' sets the percentage of envelope modulation.

Figure 4-XXI Arturia Minimoog filter's main parameters

Setting the filter resonance to its maximum makes the filter self-oscillate. Imagining a high resonance filter frequency response it is possible to visualize an increasing peak as Q increases. If this peak increases enough, the frequency response becomes an impulse (ideal situation): one impulse is placed at the cut-off frequency.

One pure tone, a sine wave, appears when the filter self-oscillates. It can be heard and it is possible to play with harmonically and melodically.

The 'AMOUNT OF CONTOUR' knob (Figure 4-XXII) sets the effect of the ADS envelope modulation (Attack, Deckay, Sustain). FulSynth uses ADSR envelope generators. Here there is no Release knob.

Figure 4-XXII ADS Minimoog envelope generator

However, Arturia Mnimoog's Release time is internally set to match the Decay's variable length. It is also possible to fix a Release time of practically 0 with a switch ("Decay") placed next to the keyboard.

Minimoog offers another cut-off frequency modulation. If the first of the three switches is activated the cut-off frequency is modulated by its keyboard modulation wheel.

Furthermore, the other two switches offer the possibility of different combination of key following modulation. Depending on their four possible respective positions the cut-off frequency is going to follow with varying precision the key pressed by the performer, so the filter will automatically follow the melody.

4.4. TEST3 – Amplification

4.4.1. Amplification Blocks

It is common not to recognize a clear amplifier block or area as easily as one could recognize the oscillator bank or the filter stage, both of which have plenty of knobs or switches. The master volume mostly defines the amplifier block but sometimes another few parameters might go with it. In FulSynth the ADSR amplitude envelope generator is placed beside the master volume. It is normal to find an amplitude envelope generator in many devices. Despite the filter envelope generator

is often placed by the filter controls, the amplitude envelope generator might be separate from the master volume, causing an initial disorientation.

Figure 4-XXIII Amplifier in each synthesizer- Bass Station (top)- Minimoog (middle)-ABL Pro (bottom)

4.4.2. Amplifying

The main output signal amplitude has been measured at different frequency input signals. The filter is completely open in each device and its resonance set to the minimum value. An array of waves is sent to the main output: different frequency square waves. The master volume for each virtual device it is set to the maximum, it keeps this value constant during the whole experiment. The volume is at its maximum to avoiding distortion. The array starts with low frequency waves and progressively increases the frequency.

The measures are carried out with AudioXplorer (Arizona Software 2006). A generic audio editor has also been used to shape the waves and illustrate the results from another perspective.

1. **Novation Bass Station** 1.5 (Novation 2007)

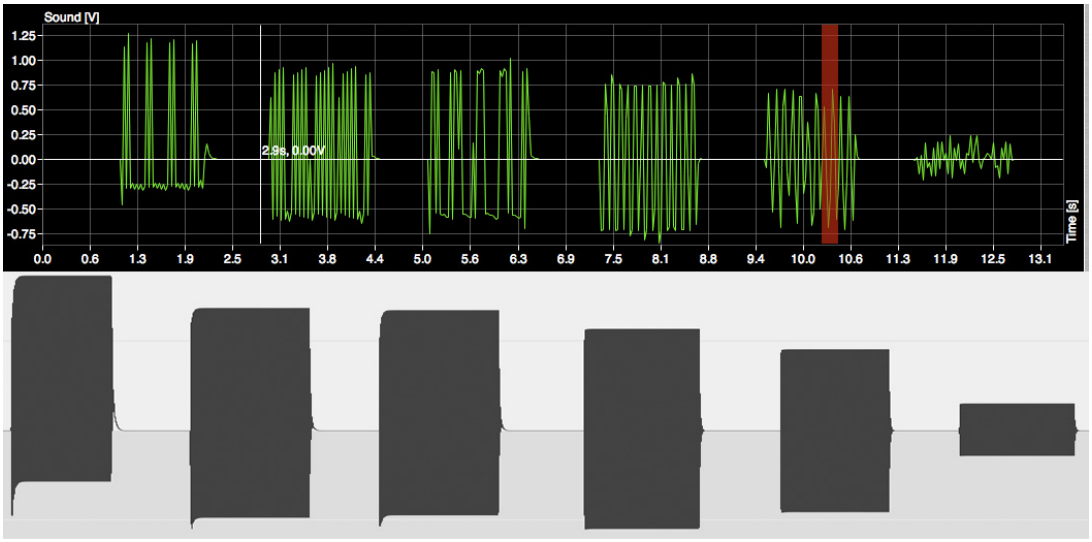


Figure 4-XXIV Novation Bass Station - Amplitude variation depending on frequency

2. **Arturia Minimoog** 1.6 (2006) (Arturia 2009)

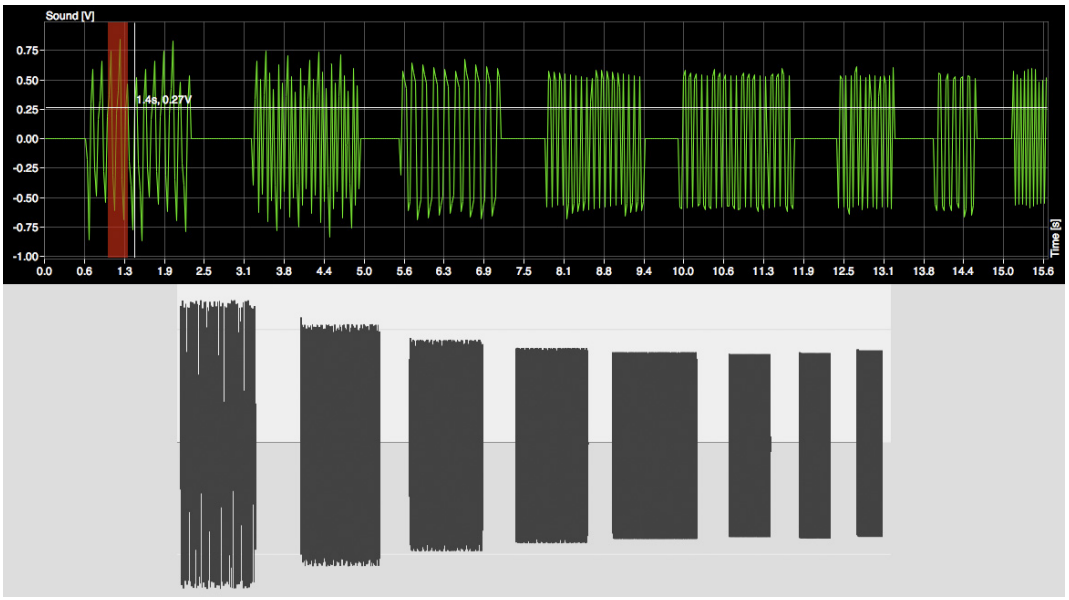


Figure 4-XXV Arturia Minimoog - Amplitude variation depending on frequency

3. **AudioRealism Bass Line Pro** 1.102 AU Intel (ABL Pro) (Realism 2009)

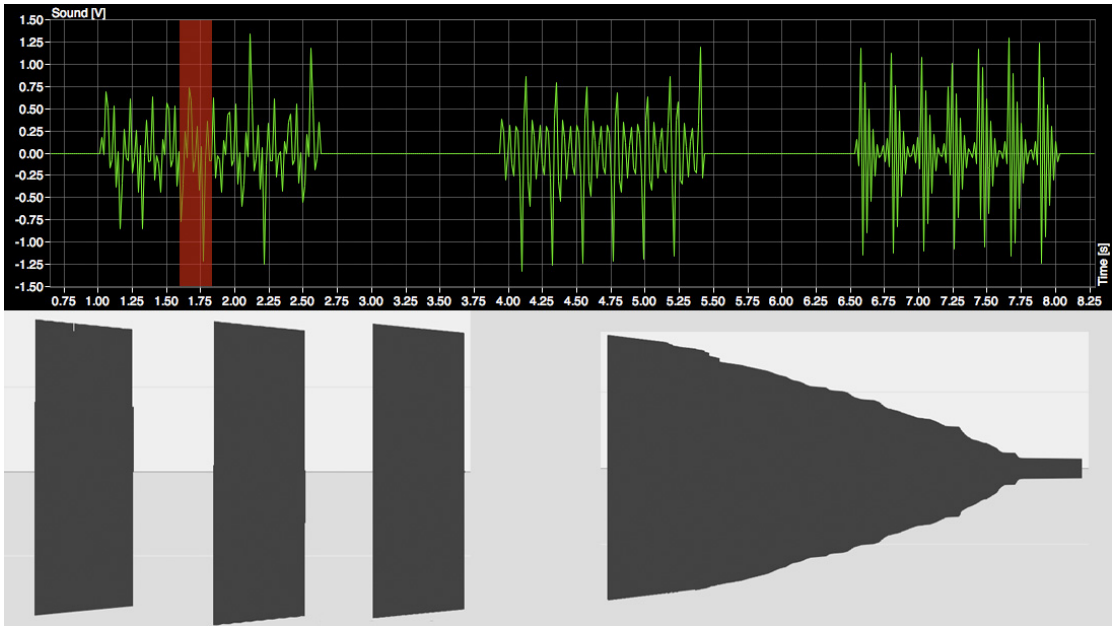


Figure 4-XXVI AudioRealism Bass Line Pro - Amplitude variation depending on frequency

4. FulSynth

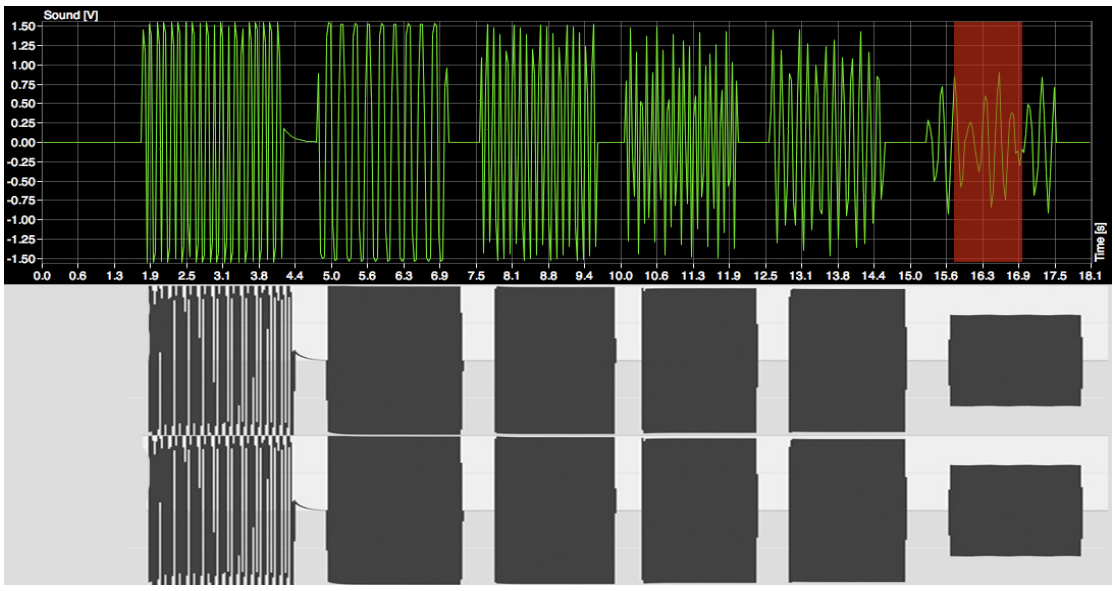


Figure 4-XXVII FulSynth - Amplitude variation depending on frequency

4.4.3. Preliminary conclusions

The frequencies in the array are between approximately 50Hz and 5kHz. All four virtual devices take into consideration the human ear's frequency response to balance the volume. This means that low frequency waves are amplified more than high ones (into the considered range). The human ear frequency response (Group Technologies 2010) is shown:

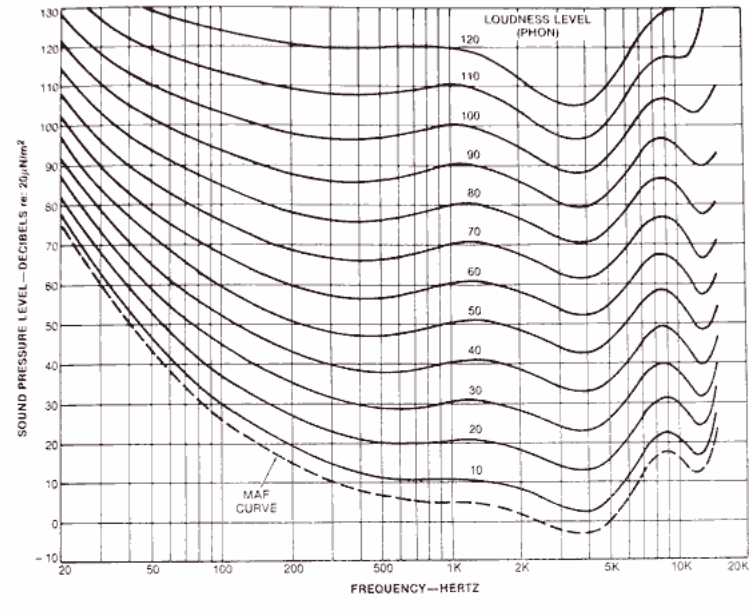


Figure 4-XXVIII Human ear frequency response

To perceive a loudness of 10PHON the sound pressure level of a 20 Hz wave has to be of almost 80dB. To perceive 10PHON from a 1kHz sound wave the sound pressure level must be 10dB.

The test sound wave graphic representations above are not detailed enough. They do not illustrate the amount of dB each device gives to each frequency. However, it is possible at least to appreciate that they all take the human ear frequency response into consideration. They all consider that the user must be able to hear low or high pitches comfortably.

FulSynth fixes the amplification of its frequencies. The procedure to determine the right level for each possible pitch has been the simplest: listening to many different pitches of different waves the amplification has been increased or

decreased depending on hearing perception. Novation Bass Station's array presents an offset in its low frequency waves, while the last three pitches are aligned.

AudioRealism Bass Line Pro's array (Figure 4-XXVII) offers a poor graphic representation in AudioXplorer. Actually, the three first pitches seem to have the same amplification in both graphics. However, during the recording process it was possible to monitor some dB decreases. Through the frequency sweep it has been easier to show how ABL Pro also considers the human ear response.

Finally, Arturia Minimoog shows up a smooth and user friendly wave shape. The components of its array are not only more amplified in low frequencies but it is also possible to recognize a small increase (the one around 4kHz, Figure 4-XXIX) when the pitches get higher. All waves are offset free.

The four devices have different effective power. Novation Bass Station and AudioRealism Bass Line Pro are able to generate louder sounds, and their volume was not completely set to the maximum the device could offer during the test. It was the maximum volume without distortion. Arturia Minimoog uses the whole knob volume range to produce a maximum amplitude wave slightly lower than the ones produced by the last two, and when its volume is set to the maximum the output presents no distortion.

FulSynth has the highest voltage output wave. However, it always sounds much lower than the other three.

4.4.4. Comparisons: AudioRealism Bass Line Pro

Figure 4-XXIX ABL Pro amplification panel

Three knobs make up the amplification stage: 'BOOST', 'SHAPE' and the master 'VOLUME'. Beside this, there is a picture of the 'VEG' envelope generator (Figure 4-XXX). However, ABL Pro places the envelope generator apart. 'VEG' it is

hard-wired to the master volume and can also be user-wired to perform other functions.

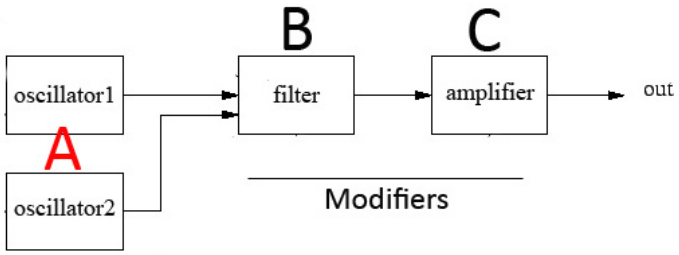
'BOOST' amplifies the low frequency content of the signal, 'SHAPE' reshapes the signal, giving body and brightness to it and the ADSR envelope generator produces the most remarkable differences with FulSynth.

FulSynth presents in its amplification panel one master volume control and one ADSR envelope generator. Its ADSR presents shorter Attack, Decay, Sustain and Release times. The Release time is especially longer in ABL Pro, when it is set to its maximum value the pitch pressed never attenuates itself. This function can be performed also in FulSynth activating the 'drag' mode at the oscillation panel (section 3.3).

CHAPTER 5. CONCLUSIONS AND FUTURE WORK

5.1. Theoretical conclusions

The subtractive synthesizers have a common core: the source-modifier method. It is essential to recognise in each device: the sound source panel (oscillators or noise), the filtering panel and the main out panel.



There are many tools a synthesizer might include to manipulate sound. FulSynth incorporates the following: the ADSR envelope generators, a low frequency oscillator modulating the low-pass filter cut-off frequency (MOOG Unit), the Delay Unit and a rhythm generator (LFO Unit).

After understanding how commercial synthesizers implement its tools, the FulSynth block diagram will be represented in the following way:

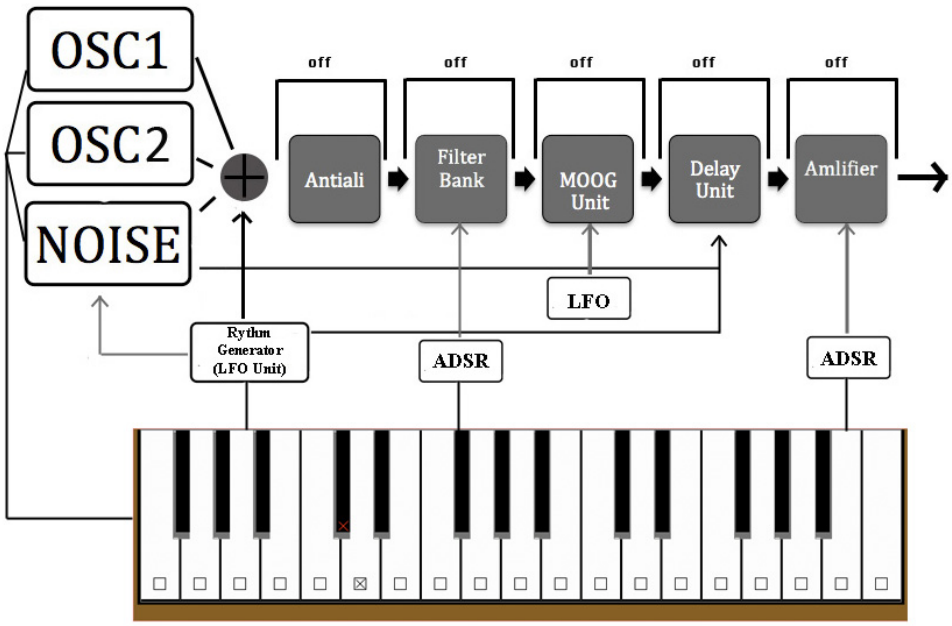


Figure 5-I FulSynth block diagram remade

The subtractive synthesis source-modifier method is the core of a subtractive synthesizer, but it is important to highlight how the method is used. In order to modify the timbre through subtraction a synthesizer might use different methods and all of them are essential complements to the subtractive synthesis fundamentals. Modulation is one of these essential concepts. It is a requisite to understand it in order to comprehend the subtractive synthesis method of producing sound. Modulation always has the same definition. However, it produces completely different results when one modulates amplitude or frequency values, when one uses an ADSR function or a low frequency oscillator. It is important to understand the concept and to recognize by hearing the effect it creates.

Controlling subtractive synthesis method understanding its related tools will allow the performer to truly control the instrument and eventually a sound designing process will follow defined steps.

5.2. Practical conclusions

The FulSynth **oscillator block** is able to produce eight different waves and to tune them in a different octave range. However, to include a "detune" control could be useful to emulate a wider range of sounds or to include a modulation feature.

The FulSynth ADSR **filter bank** subtracts the unwanted frequencies, but an increase in the order of the filters would give a proportionately a sharper slope, and that would allow the modifying of timbre with greater detail.

The **antialiasing** filter cuts-off high frequency noise. However, low frequency interference occurs while playing high frequency waves with a rich harmonic content (sawtooth, square, sawtri).

The **ADSR envelope generators** work properly, but the envelope generators used by the other devices control the signal with more precision and also have a wider range of values.

The **inner-communication** works as expected, but it could be improved. The three analysed devices do not present clipping in any situation and the main signal flow is never cut down by the action of any switch. The FulSynth ADSR amplifier or

the Delay Unit might create unwanted situations when they are switched, cutting down the main signal flow.

The FulSynth **amplification** is balanced. It takes into consideration the human ear's frequency response. However, volume conflicts might appear when all tools are interacting, especially when the four possible sources are active. Both oscillators alone are efficiency reproduced, but when the Noise Generator Unit and the LF-Rhythm Generator are added the volume of the oscillator decreases and the main output signal sounds too low and unclear.

The amplitude issues might also be related to the inner-communication design. FulSynth's signal flows from one abstraction to another through the *through~/catch~* Pd functions. The amplitude seems to decrease when the channel these functions create carries too much information. However, it is not only because of this that a Pd-developed synthesizer seems to be less efficient in amplitude terms. In Chapter 4, section 4.4.4, the graphics showed the amplitude level in Volts obtained for every device. FulSynth obtains the highest level (Figure 4-XXVIII), close to 1.5V, but when it plays at its maximum volume the sound reproduced by the soundcard is much lower that the level obtained with any of the other three. The use the commercial analysed devices make of the soundcard is much more efficient.

In terms of wave shaping, filtering and amplifying Arturia Minimoog gives the best readings. However, all three devices include features that would improve FulSynth considerably.

5.3. Future work

In Pd development terms it is absolutely possible to improve the FulSynth programming. The FulSynth filters and envelope generators are for instance not self-designed functions. An advanced Pd developer using electronic engineering concepts might obtain better filtering results. Further learning in this language will create a better FulSynth Pd synthesizer. It will be important to learn more functions and procedures, but it will be essential to understand better the Pd-computer communication and to see where Pd's limitations exist. I

FulSynth presents important low frequency interference when rich harmonic content high frequency waves are reproduced. However, the FulSynth antialiasing filter attenuates properly the high frequency digital noise in these waves. It is essential to do a further analysis in order to assure the reason of this interference.

In terms of physical improvement, there are many features that could be added in order to create a more complete virtual synthesizer: modulation options, patching possibilities, bank of presets, etc. A first improvement could be for example to program FulSynth as a polyphonic instrument.

Another point to develop further is the user perspective. Performers and sound-designers especially understand the instrument's advantages and limitations. It will be always essential to listen to their opinions in order to develop better products. However, personal skills in performance and sound designing are always to be considered. Developing this researching and programming project has not left enough time to experiment with Fulsynth. In order to improve its features it will be of vital importance to play with it more and also with other synthesizers.

REFERENCE LIST

Abdullah, T. Y. (Blog [online]). "Angelfire, Synthesizers, Music & Television [online]." Retrieved 20 Jan, 2011, from www.angelfire.com/in2/yala/2ansynth.htm.

Angus, D. M. H. a. J. (1996). Acoustics and Pychoacoustics.

Anonymous. "ADSR (FILTER)." Retrieved 31 Jan, 2011, from http://www7a.biglobe.ne.jp/~mpc1000/os2/adsr_filter_e.htm.

Anonymous. ([online]). Retrieved 28 Jan, 2011, from http://4.bp.blogspot.com/_WYOveW87pf4/TOuSea6VCKI/AAAAAAAAAVY/KKueY2UzX6U/s400/low_pass_filter.gif.

Anonymous. (Blog [online]). "Historia de los Instrumentos Electronicos." from http://electroinstrumentos.blogspot.com/2010/07/moog-synthesisers-robert-moog-usa-1963.html.

Arizona Software (2006). AudioXplorer: free licence.

Arturia. (2009). "Minimoog V2 English Manual." Retrieved 28 Jan, 2011, from http://www.arturia.com/evolution/en/downloads/manuals.html.

BMARS, B. M. A. R. S. D. C. (Blog [online]). "Filters." Retrieved 20 Jan, 2011, from http://eamusic.dartmouth.edu/~book/MATCpages/chap.4/4.3.filt.html.

Cambridge, U. P. (2010). "Cambridge Dictionaries Online." Retrieved 20 Jan, 2011, from http://dictionary.cambridge.org.

Cassidy, R. J. a. S., Julius O., Center for Computer Reearch in Music and Acoustics (CCRMA), Stanford University. (2008). "What is a Filter Bank?" Retrieved 20 Jan, 2011, from https://ccrma.stanford.edu/realsimple/aud_fb/What_Filter_Bank.html.

En el tintero de Zamzara. (2006, 2 Oct 2008). "¿Por qué los teclados no están en orden alfabético? QWERTY vs. DVORAK." Retrieved 3 Feb, 2011, from http://zamzaraopina.wordpress.com/2006/08/29/¿por-que-los-teclados-no-estan-en-orden-alfabetico-qwerty-vs-dvorak/.

eNotes. (2011). "Butterworth filter." Retrieved 28 Jan, 2011, from http://www.enotes.com/topic/Butterworth_filter.

Gentry, K. a. K., D., University of Illinois. (2008). "BIOE Circuits - Steady-State Analysis." Retrieved 20 Jan, 2011, from http://courses.engr.illinois.edu/bioe415/tutorial/ac/ac.html.

Group Technologies. (2010). "The Birth of the Array: Part Two." Retrieved 12 Feb, 2011, from http://www.gtaust.com/filter/05/07.shtml.
Jenkem, D. (2009, 14 Sep 2010). "How to make a wobble bass with Reason." Retrieved 28 Jan, 2011, from http://djjenkem.wordpress.com/2009/04/16/how-to-make-a-wobble-bass-with-reason/.

John Hopkins University and Ross, M. (2010). "Listen to Fourier Series [online]." Retrieved 20 Jan, 2011, from http://www.jhu.edu/signals/listen-new/listen-newindex.htm.

Kettlewell, B. (2002). Electronic Music Pioneers.

Kreidler, J. (2009). Chapter 3. Audio. Programming Electronic Music in Pd -. M. U. o. Freiburg, http://www.pd-tutorial.com/.

LoveToKnow. (2011). "Your Dictionary." Retrieved 6 Feb, 2011, from http://www.yourdictionary.com/modulation.

Lundberg, K. H., Massachusetts Institute of Technology (MIT). (2002). "So you want to build an analog synthesizer?" Retrieved 28 Jan, 2011, from http://web.mit.edu/klund/www/weblatex/node2.html.

Making Music. (2010). "Quick Guide To Envelopes." Retrieved 28 Jan, 2011, from http://www.making-music.com/envelopes.html.

Matlab (2008). Matlab R2008a.

Matrixsynth, B. (2010). Retrieved 28 Jan, 2011, from http://matrixsynth.blogspot.com/2010/03/aelita-demo-by-syntezatoryprvpl.html.

Novation (2007). Bass Station, virtual bass synthesizer: User manual and download at link.

Peters, C. C. Retrieved 28 Jan, 2011, from http://www.cchrispeters.com/Images/EnvelopeGenerator.gif.

Pressing, J. (1992). SYNTHESIZER PERFORMANCE and Real-Time Techniques.

Puckette, M. (2006). The Theory and Technique of Electronic Music World Scientific Publishing Co. Pte. Ltd.

Puckette, M. a. P. c. (1996). Pure Data: Open source free software.

PunBB. (2005). "Pure Data Forum~." Retrieved 09 Feb, 2011, from http://puredata.hurleur.com/sujet-1705-achieving-24db-oct.

Radio-Electronics. "Quartz crystal band pass filters." Retrieved 28 Jan, 2011, from http://www.radio-electronics.com/info/receivers/filters/crystal_bandpass_filters.php.

Realism, A. (2009). Bass Line Pro: Sequenced synthesizer, plug-in.

Russ, M. (1996). Sound Synthesis and Sampling (2nd Edition), Focal Press.

Samplecraze. (2010). Retrieved 28 Jan, 2011, from http://www.samplecraze.com/images/july/synthesis5_clip_image002.jpg.

Schoenfelder, V. f. t. D. o. S. E. a. T. C. S., Technische Universität Berlin. (2009, 16 Feb 2009). "Unveiling Tone-in-Noise detection." Retrieved 28 Jan, 2011, from http://www.cognition.tu-berlin.de/menue/research/projects/tone-in-noise_detection/.

Sievers, B. "The Amateur Gentleman's Introduction to the Principles of Music Synthesis." Retrieved 28 Jan, 2011, from http://www.beausievers.com/synth/synthbasics/.

Strange, A. (1972). Electronic Music. Systems, techniques, and controls.

Tillman, D. J. (2006, September 2006). "Moog patents." Retrieved 28 Jan, 2011, from http://www.till.com/articles/moog/patents.html.

Vidal Reynés, G. (2000). "El sintetizador " Retrieved 20 Jan, 2011, from http://personal.telefonica.terra.es/web/gvre/CARS/htm/sintetizador.html.

Vintage Synth Explorer. (2011). "Roland SH-3a." Retrieved 28 Jan, 2011, from http://www.vintagesynth.com/roland/sh3.php.

Wikipedia. (2010, 22 Dic 2010 at 11:20). "Electronic filter topology." Retrieved 28 Jan, 2011, from http://en.wikipedia.org/wiki/Electronic_filter_topology.

Wikipedia. (2010, 15 December 2010 at 19:22). "Low-pass filter." Retrieved 28 Jan, 2011, from http://en.wikipedia.org/wiki/Low-pass_filter.

Wikipedia. (2011, 27 Jan 2011 at 21:27). "Minimoog." Retrieved 31 Jan, 2011, from http://en.wikipedia.org/wiki/Minimoog.

Wikipedia. (2011, 28 Jan 2011). "Nyquist–Shannon sampling theorem." Retrieved 28 Jan, 2011, from http://www.arturia.com/evolution/en/downloads/manuals.html.

Wikipedia. (2011, 29 January 2011 at 16:29). "Nyquist–Shannon sampling theorem." Retrieved 2 Feb, 2011, from http://en.wikipedia.org/wiki/Nyquist–Shannon_sampling_theorem.

Wikipedia. (2011, 4 Jan 2011 at 23:54). "Poisson summation formula." Retrieved 2 Feb, 2011, from http://en.wikipedia.org/wiki/Poisson_summation_formula.

Wikipedia. (2011, 16 Jan 2011). "Roland SH-3A." Retrieved 28 Jan, 2011, from http://en.wikipedia.org/wiki/Roland_SH-3A.

Williams, C. S. (1986). Designing Digital Filters.

Appendix A

FulSynth Filters

1. FulSynth filters

FulSynth has four different filters (see sections 3.5, 3.6) implemented through two Pure Data components: *svf~,* which implements the three different ADSR modulated filters included in the 'FILTERBANK' unit, and *moog~,* implementing the low frequency modulated low-pass filter in the MOOG Unit (see functions in appendix).

In FulSynth *svf~* working as a low-pass filter and *moog~* have been analyzed using AudioXplorer (Arizona Software 2006).

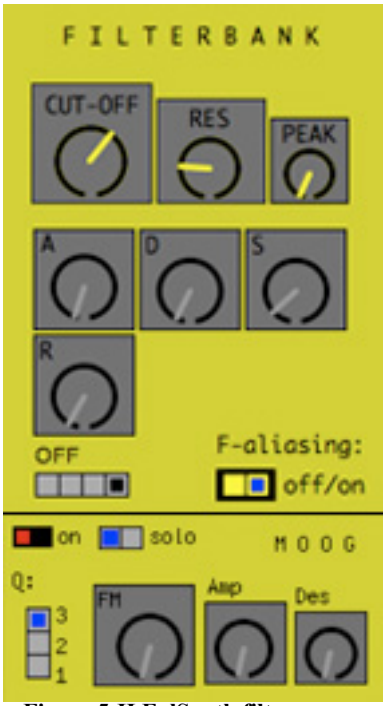


Figure 5-II FulSynth filters

Svf~ 's Pd component developer doesn't give (in the component help file) the slope value of this multi-band choice filter. However, some posts in Pd online forums talk about a value of 12dB/oct (PunBB 2005). Besides, an emulation of the classical Moog filter should have a 24db/oct slope.

2. Analysing FulSynth *svf~* low-pass filter

To define a filter it is essential to visualize the module of its frequency or impulse response. A *dirac~* object generates an impulse in Pd. Applying a delta function into the filter's input it is possible to obtain the necessary response.

For each measurement one delta function has been Pd generated and filtered with FulSynth.

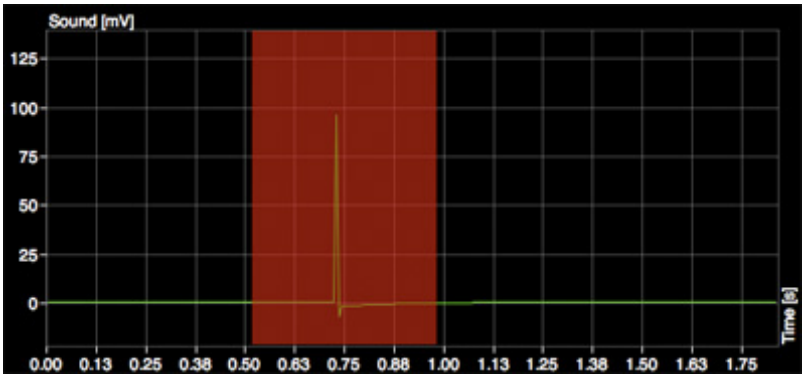


Figure 5-III Impulse Pd generated

During this first experiment many impulses were sent to the filter in order to visualize the filter response, its cut-off frequency and the slope value. Results have always set a slope value of approximately 10dB/octave. Generating the *svf~* low-pass filter frequency response:

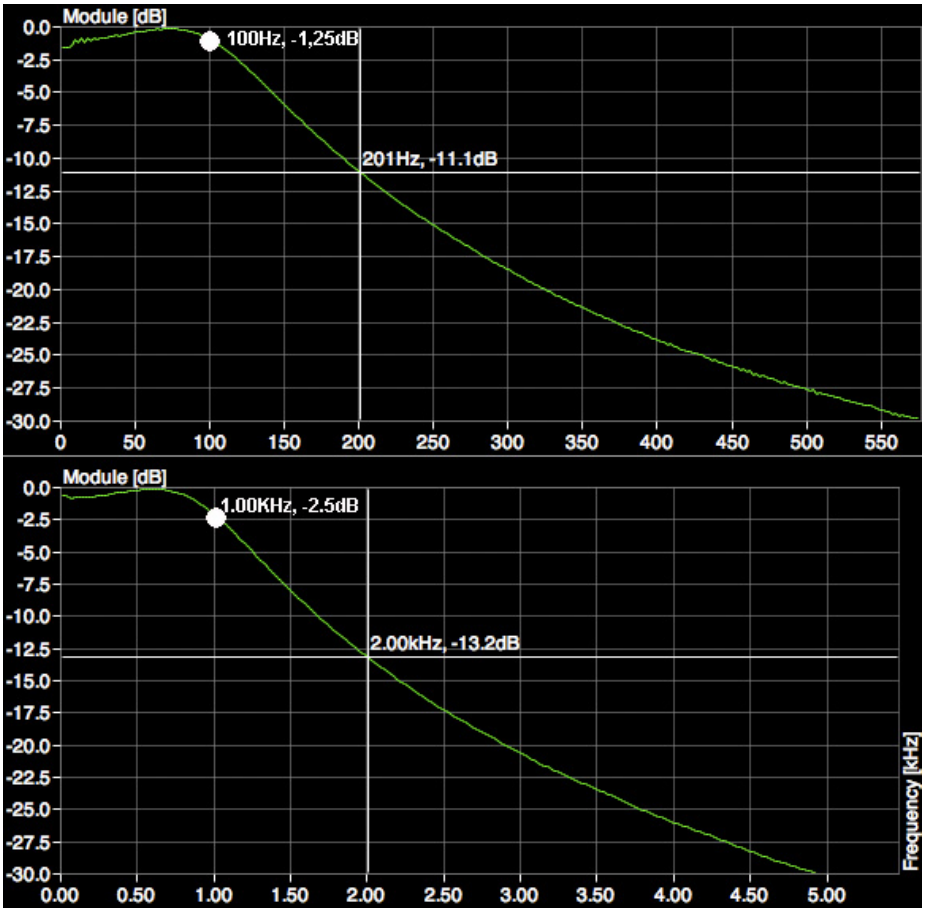


Figure 5-IV Pd *svf~* low-pass filter frequency response. Fc= 140Hz (top). Fc=1100Hz (bottom)

The graphic above (fc_{-3dB}= 140Hz, Figure 4-XVIII) sets a slope value of 9.85dB/oct while the one below (fc_{-3dB}=1100Hz, Figure 4-XVIII) sets a value of

10.7dB/oct. These are acceptable values for a low-pass filter slope. However, they could be closer to a 12dB/oct standard slope low-pass filter value. An increasing of the slope gives sharpness to the filter but it has a computational cost. 12dB/oct would be a more than reasonable value to be working with using one conventional computer.

The following measures show the influence of the resonance parameter (Q):

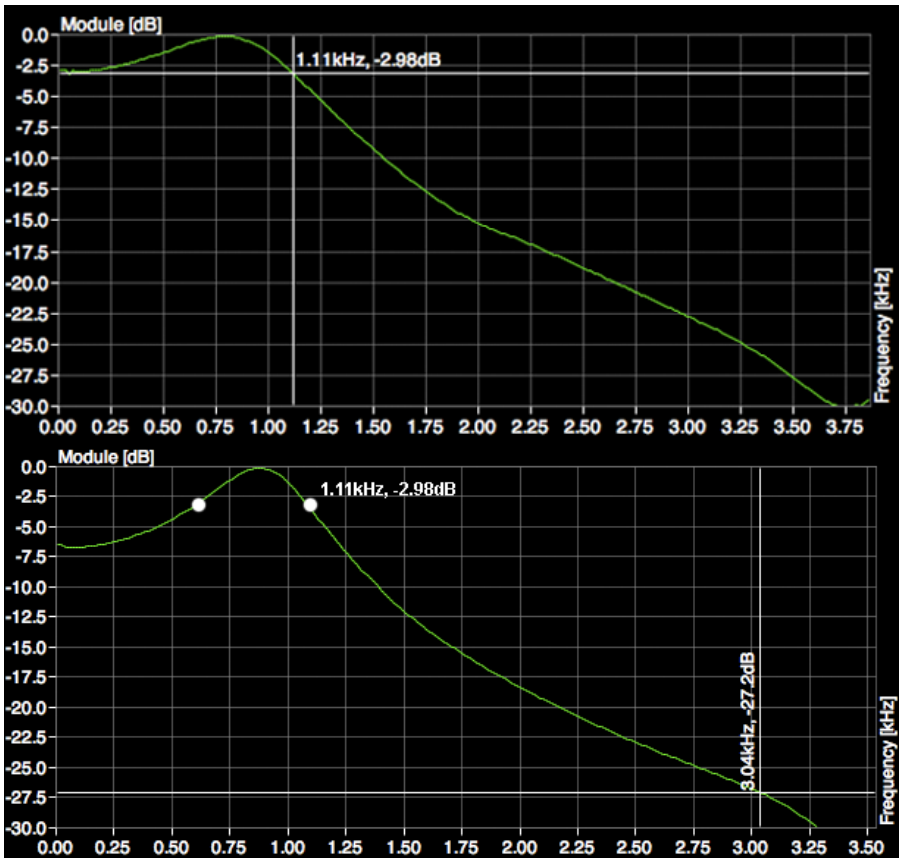


Figure 5-V Pd *svf*~ low-pass filter frequency response. Fc= 1,11kHz

Just before the cut-off frequency there is a peak where the frequency content is going to be stressed or highlighted. The graphic above has a lower resonance than the one placed below (Figure 4-XIX). It is interesting to observe how the peak increases with the resonance, and how a low-pass filter almost becomes as a band-pass filter when the resonance is high, with two cut-off frequencies at -3bB.

3. Analysing the FulSynth *moog*~ low-pass filter

The same procedure as applied in the last section has been used here to analyse the Pd *moog*~ low-pass filter frequency response. The FulSynth MOOG Unit

filter is the only one active in the device. Although the slope of the original Moog analogue filter was 24dB/oct, one analysis using AudioXplorer (Arizona Software 2006) has set an average value of approximately 18dB/oct for the *moog~* filter slope.

Showing some results:

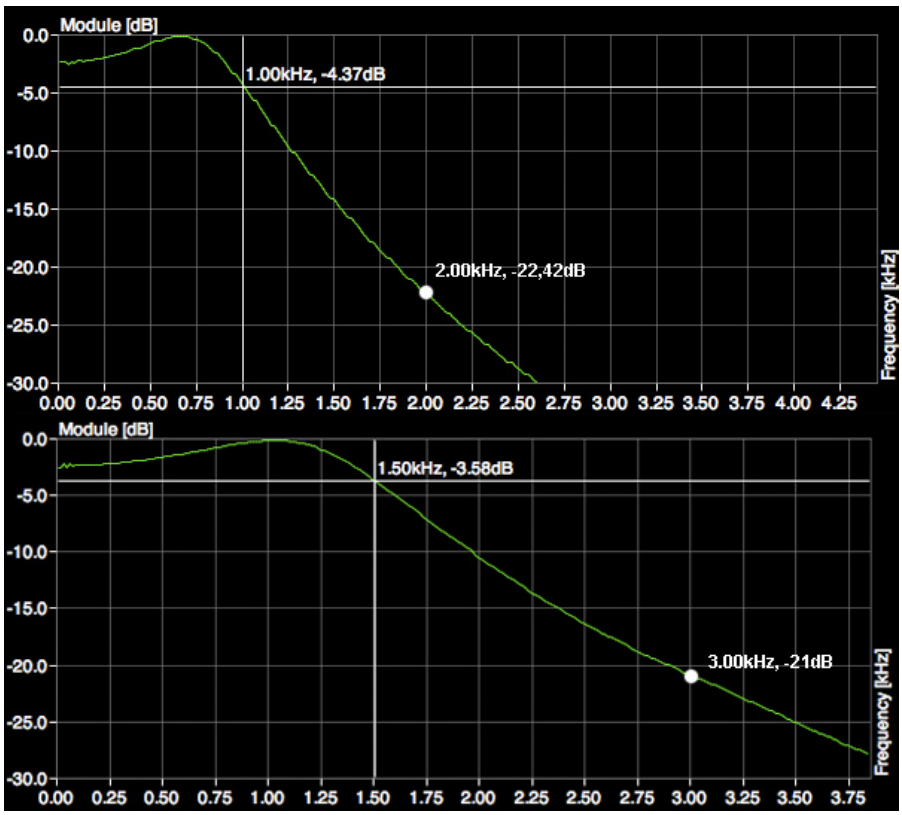


Figure 5-VI Pd *moog~* filter frequency response. Fc=900Hz (top). Fc=1400Hz (bottom)

The graphic above (Figure 4-XX) has a cut-off frequency of around 1kHz, and the value of its slope is 18.05dB/oct. The graphic below has a higher cut-off frequency, around 1.5kHz, and the value of its slope is of 17,42dB/oct.

As it has previously been mentioned, the analysis made in this dissertation is just a first reading and is not as rigorous as one done with appropriate electronic lab equipment. The goal is to highlight filters purposes and functions, to take into consideration the current results without forgetting that further analysis can always be done.

The following measures show the moog~ filter cut-off frequency modulated by the MOOG Unit low frequency oscillator (‘FM’, Figure 4-XXI).

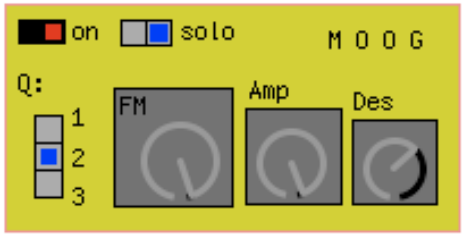


Figure 5-VII FulSynth MOOG Unit

An impulse train has been sent to the *moog~* filter input. The following are the values set in the unit:

$FM_{(LFO)}$= 3Hz

Amp=1500Hz

Des= 1000Hz

Figure 5-VIII MOOG Unit values

Looking back to the already explained theory (section 3.1.6):

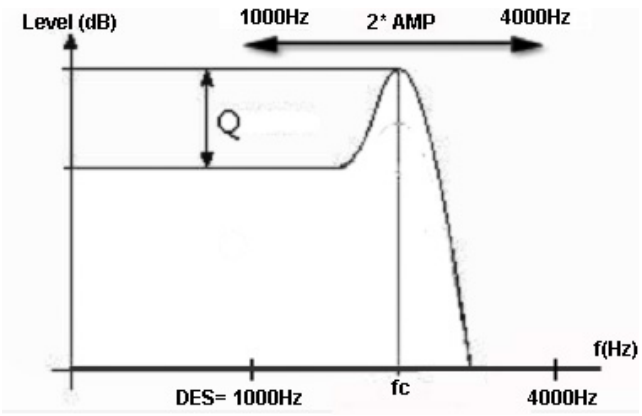


Figure 5-IX MOOG Unit cut-off frequency (fc) modulation

A train of six delta impulses has been sent to the filter input. Each delta arrives to the filter in a different time instant. Due to the modulation of the cut-off frequency through a 3Hz low frequency oscillator (three cycles per second, from 1000Hz to 4000Hz), six different fc values have been obtained; one fc for each delta and instant.

The impulse train is hardly represented due to computational problems. However, each delta has indeed produced its own frequency response. The following graphics (Figure 4-XXV) sometimes differ in the scale. It has been difficult to picture how the frequency response changes. However, analyzing the following obtained

values one can indeed see the theory explained in section 3.1.6. Taking into consideration Figure 4-XXII and XXIII, these are the values obtained:

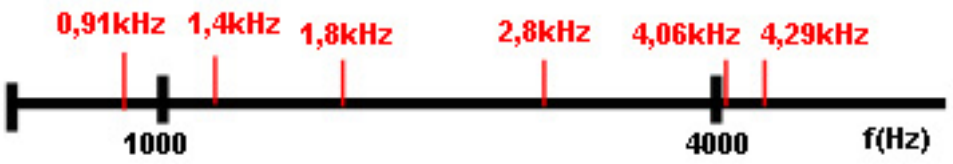


Figure 5-X Peak values for the moog~ filter frequency response

And the frequency responses for each delta of the train:

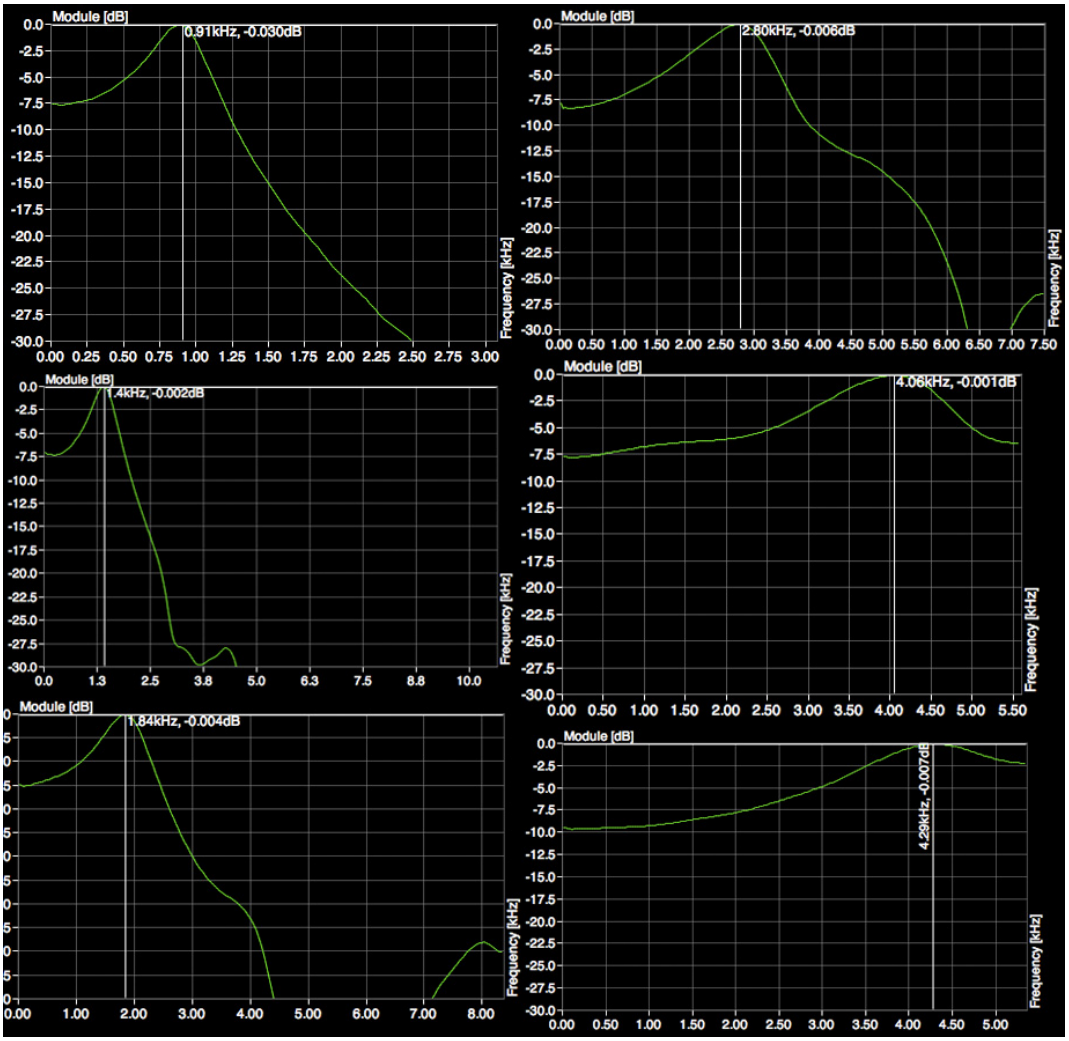


Figure 5-XI Pd *moog~* filter cut-off frequency modulated by a 3Hz low frequency oscillato

Although some values might fall out of the theoretical range (Figure 4-XXIV), it is important to notice the underlined values are *peak* values, which are really close to the fc_{-3dB}'s values. The theoretical oscillation range, which goes from 1000Hz to 4000Hz, is the range fc_{-3dB} follows. However, with this analysis it is possible to

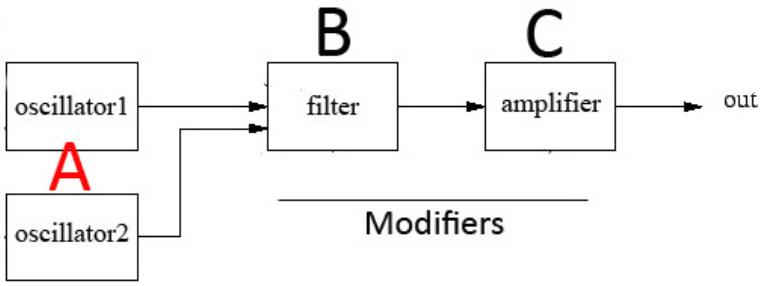
determine that the fc values will be inside the expected range. This is a graphical corroboration of the mathematical function programmed in the MOOG Unit (see function in appendix).

Appendix B

Subtractive Synthesizer Interface

1. Starting with a synthesizer

At first sight, the synthesizer's main panel can be awkward to the inexperienced user, but most subtractive synthesizers follow the already explained source-modifier structure.



This chapter provides some advice to new synthesizer users. It is essential to firstly identify the 'A', 'B' and 'C' blocs in any subtractive synthesizer device: the oscillators, the filter and the amplifier. Once this is done, its important to recognize where the envelope generator controls are placed. They might be a part of the block they are going to affect, but this is not the only possibility. Envelope generators might be used to modulate filter frequency, final amplitude or even oscillator signal source parameters. They can appear separately and are another distinct component of a subtractive synthesizer. Labels around them will state their function/s. Users must be aware of its location and find out its different routing possibilities.

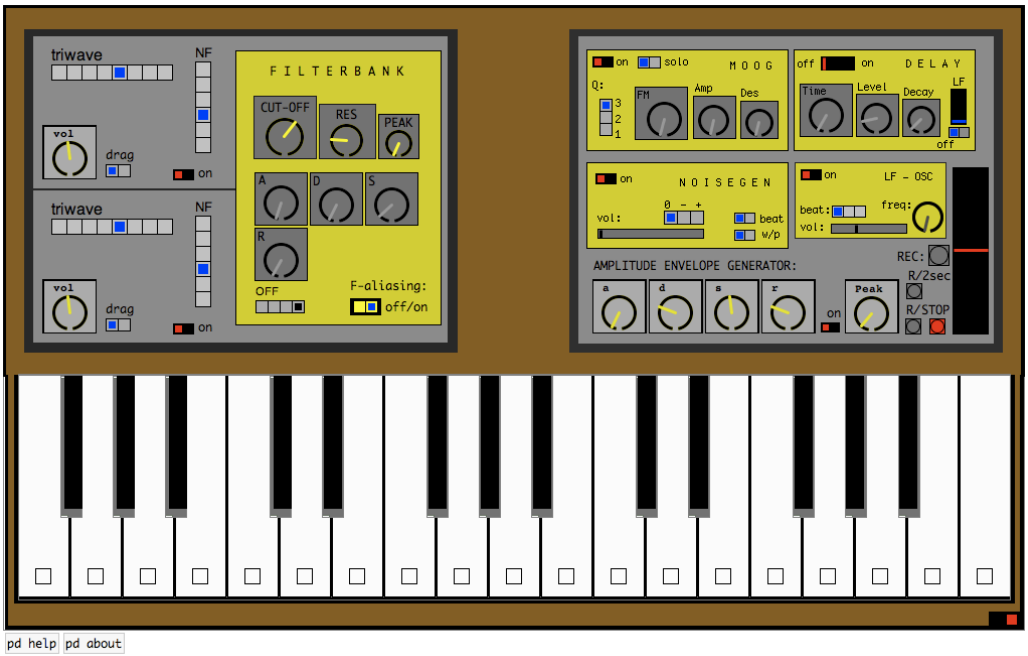
Subtractive synthesizer interfaces may include several knobs, switches, extended functionalities and unknown tools, but they all act around this first basic bloc diagram. Software-instrument user manuals will provide the performer with the necessary information to go ahead after the basics of subtractive synthesis are known.

Appendix C

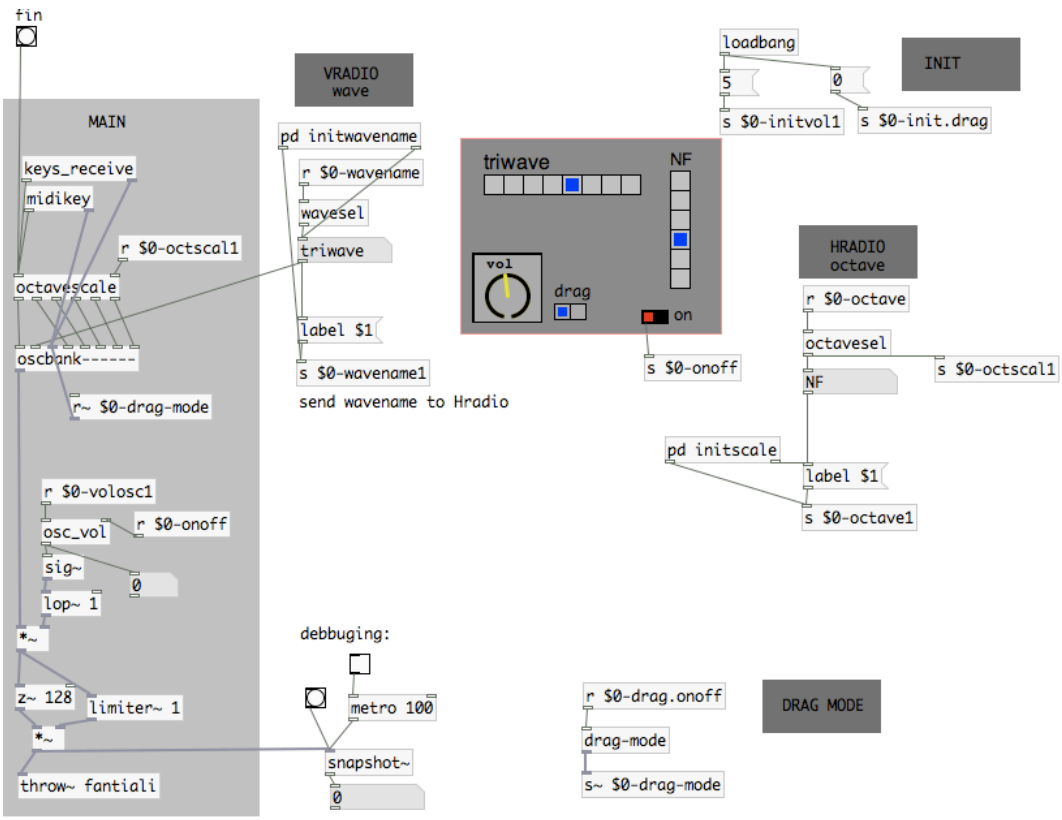
FulSynth Pure Data Functions

2. Main Interface

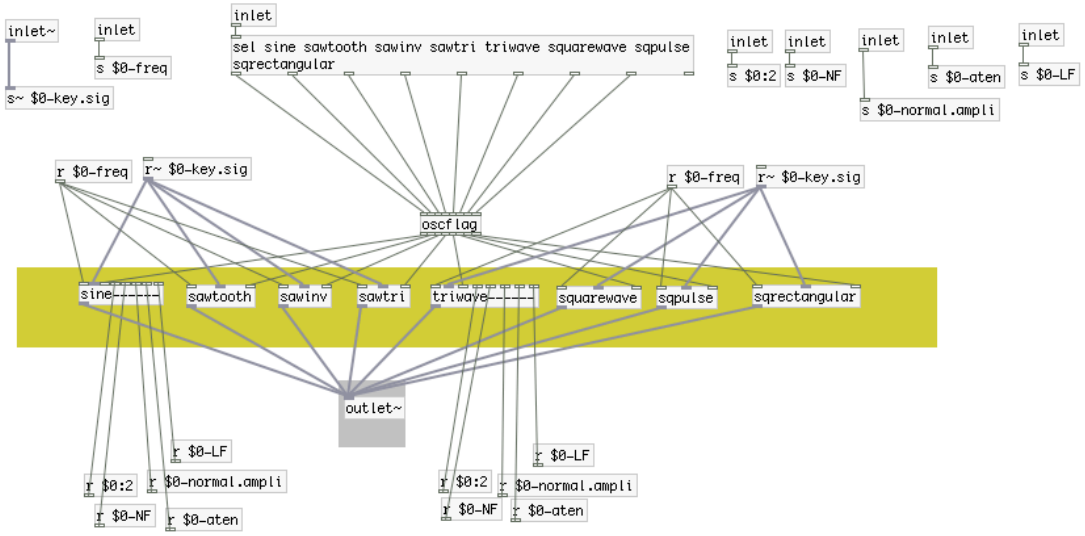
FulSynth.pd



osc1_interface.pd (oscillators one and two are an identical abstraction)



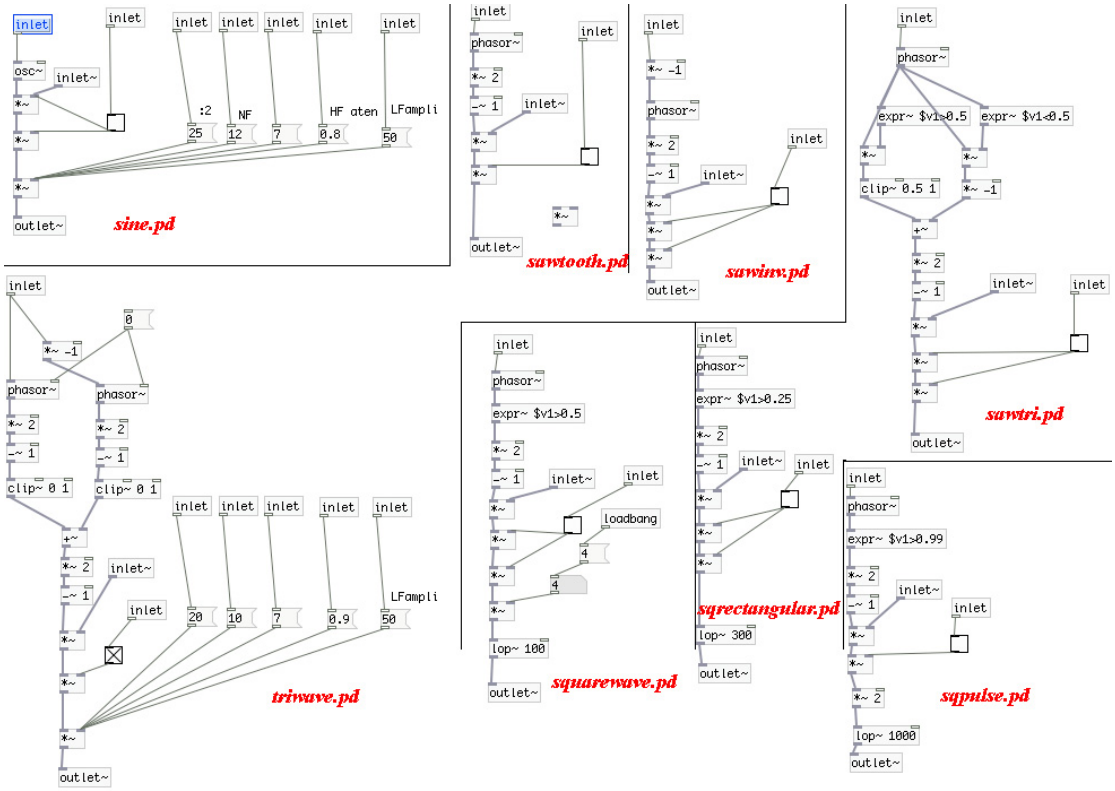
oscbank------.pd


inlet~
inlet
s $0-freq
s~ $0-key.sig
inlet
sel sine sawtooth sawinv sawtri triwave squarewave sqpulse sqrectangular
inlet
s $0:2
inlet
s $0-NF
inlet
s $0-normal.ampli
inlet
s $0-aten
inlet
s $0-LF
r $0-freq
r~ $0-key.sig
oscflag
r $0-freq
r~ $0-key.sig
sine
sawtooth
sawinv
sawtri
triwave
squarewave
sqpulse
sqrectangular
outlet~
r $0-LF
r $0:2
r $0-normal.ampli
r $0-NF
r $0-aten
r $0-LF
r $0:2
r $0-normal.ampli
r $0-NF
r $0-aten

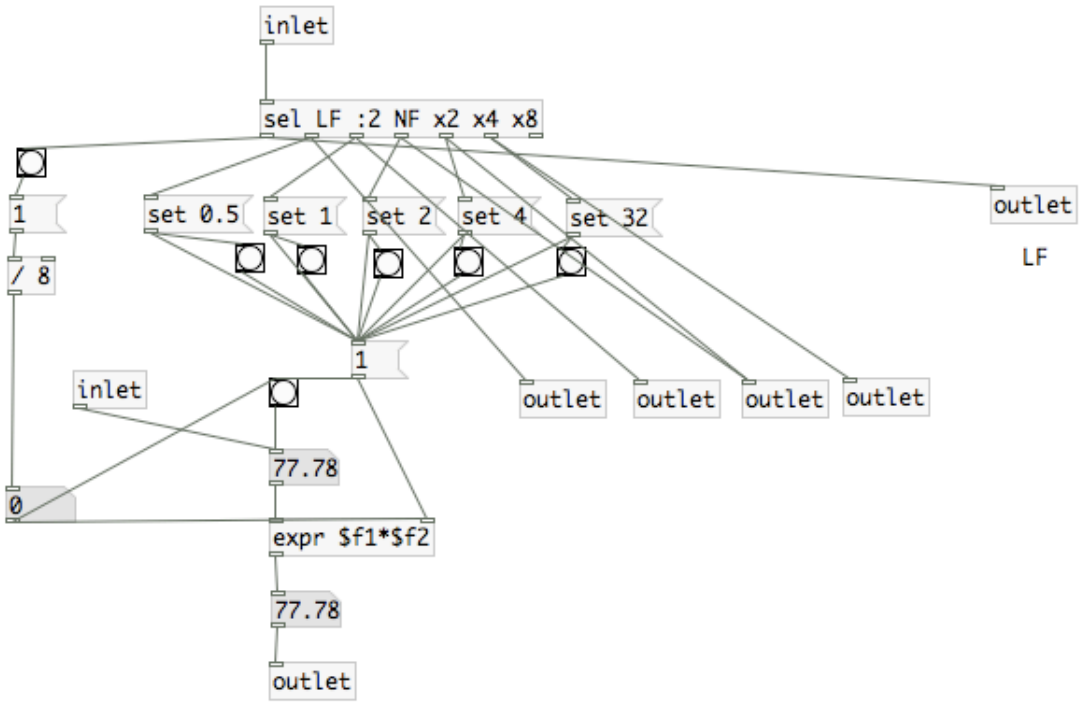
oscflag.pd

inlet
1
0
s $0-fsine
receive $0-fsqrectangular
receive $0-fsqpulse
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawinv
receive $0-fsawtooth
outlet
inlet
1
0
s $0-fsawtooth
receive $0-fsqrectangular
receive $0-fsqpulse
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawinv
receive $0-fsine
outlet
inlet
1
0
s $0-fsawinv
receive $0-fsqrectangular
receive $0-fsqpulse
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawtooth
receive $0-fsine
outlet
inlet
1
0
s $0-fsawtri
receive $0-fsqrectangular
receive $0-fsqpulse
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawinv
receive $0-fsawtooth
receive $0-fsine
outlet
inlet
1
0
s $0-fsquarewave
receive $0-fsqrectangular
receive $0-fsqpulse
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawinv
receive $0-fsawtooth
receive $0-fsine
outlet
inlet
1
0
s $0-fsqpulse
receive $0-fsqrectangular
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawinv
receive $0-fsawtooth
receive $0-fsine
outlet
inlet
1
0
s $0-fsqrectangular
receive $0-fsqpulse
receive $0-fsquarewave
receive $0-ftriwave
receive $0-fsawtri
receive $0-fsawinv
receive $0-fsawtooth
receive $0-fsine
outlet

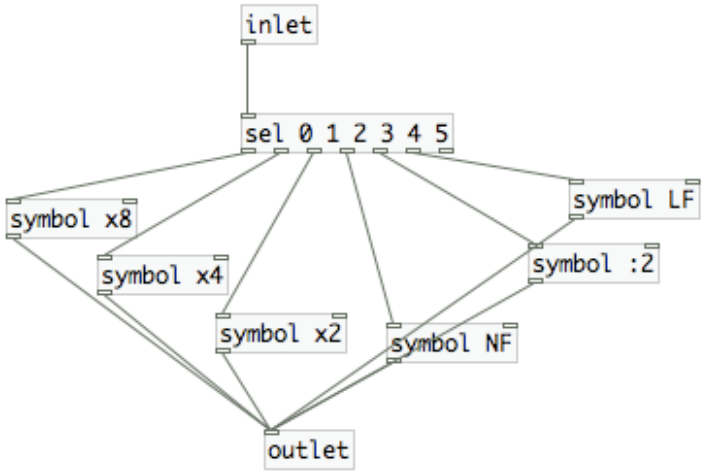
Waveshaping the eight possible FulSynth signals (eight different abstractions in one picture):



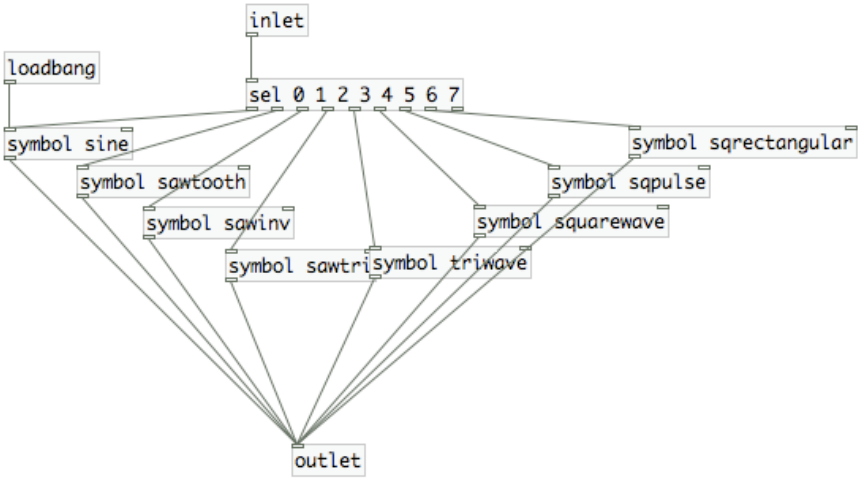
octavescale.pd



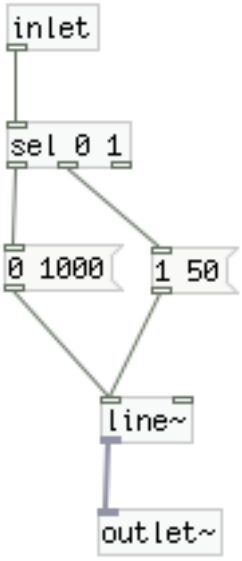
octavesel.pd



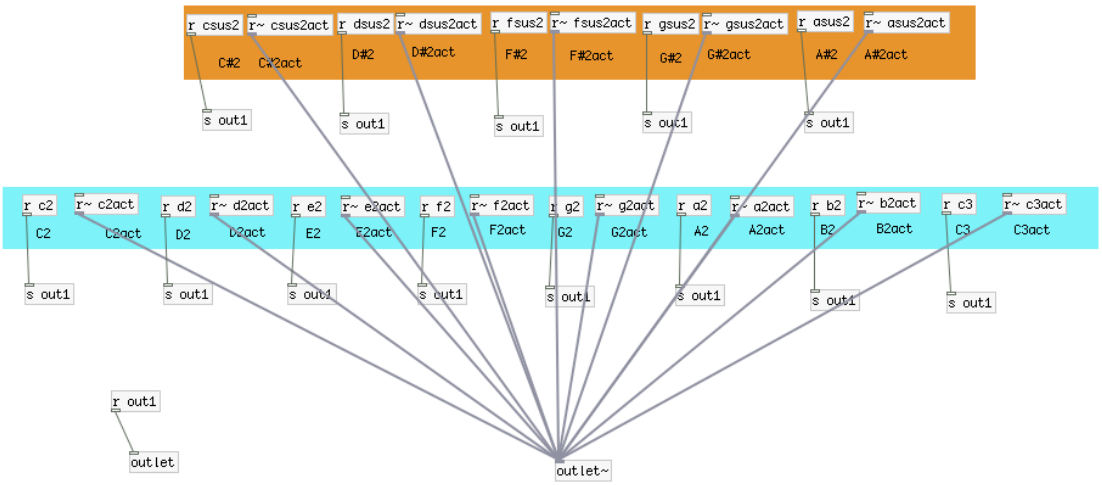
wavesel.pd



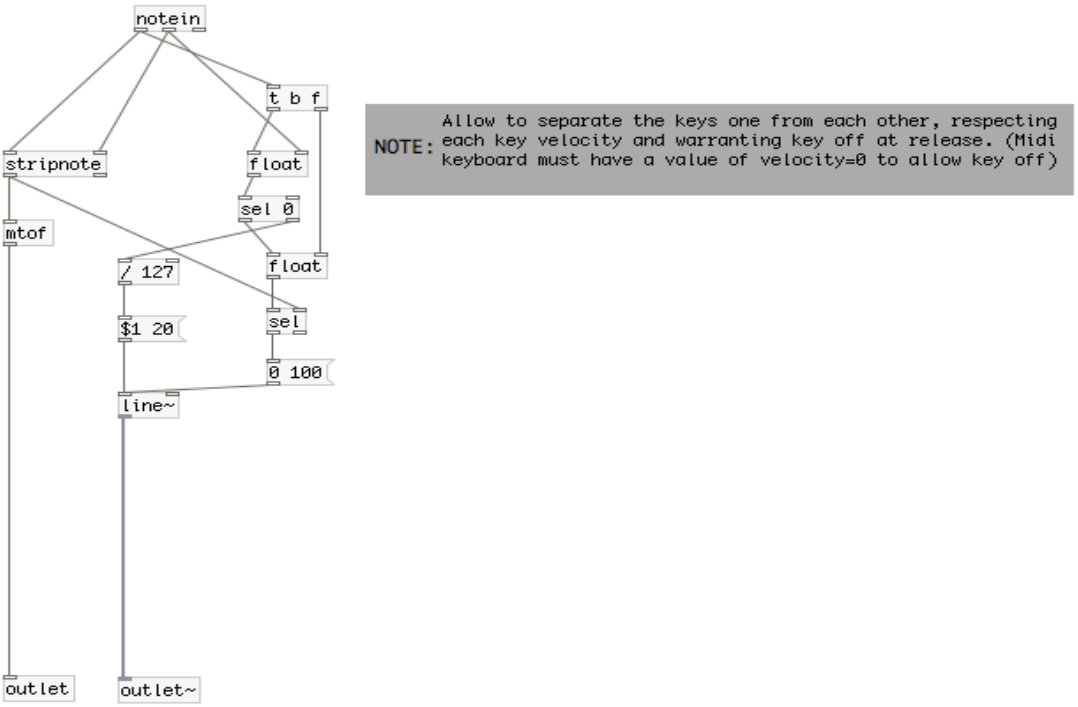
drag-mode.pd



keys_receive.pd

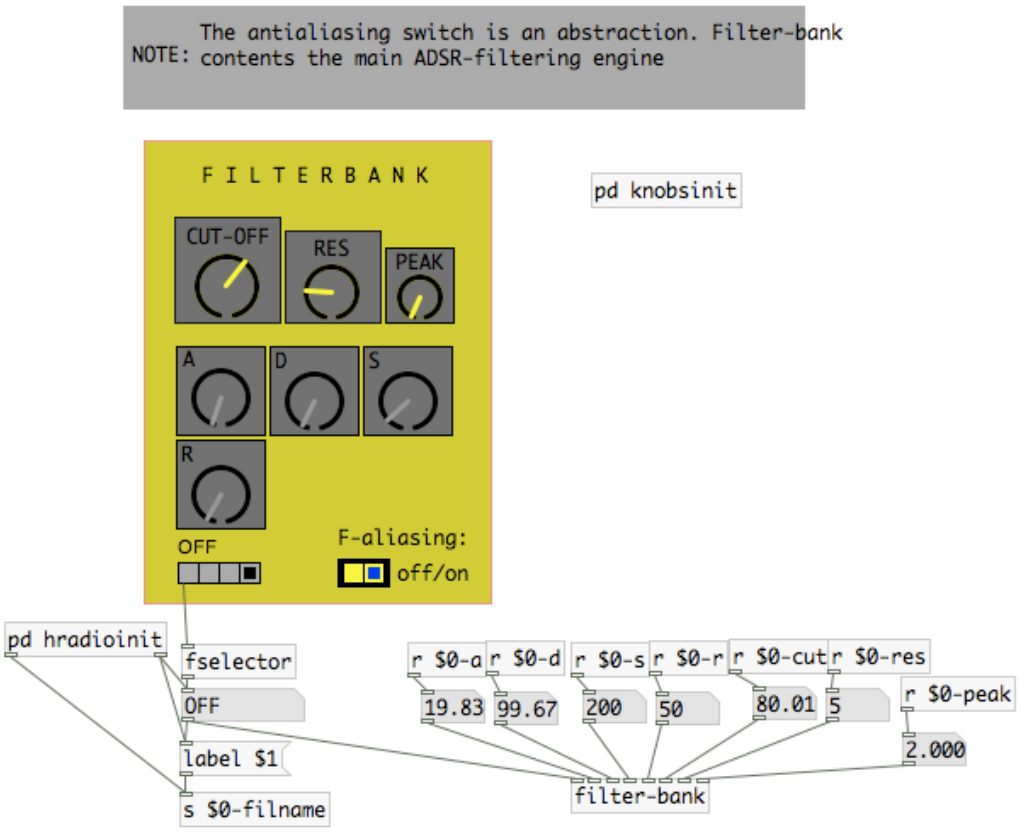


midikey.pd

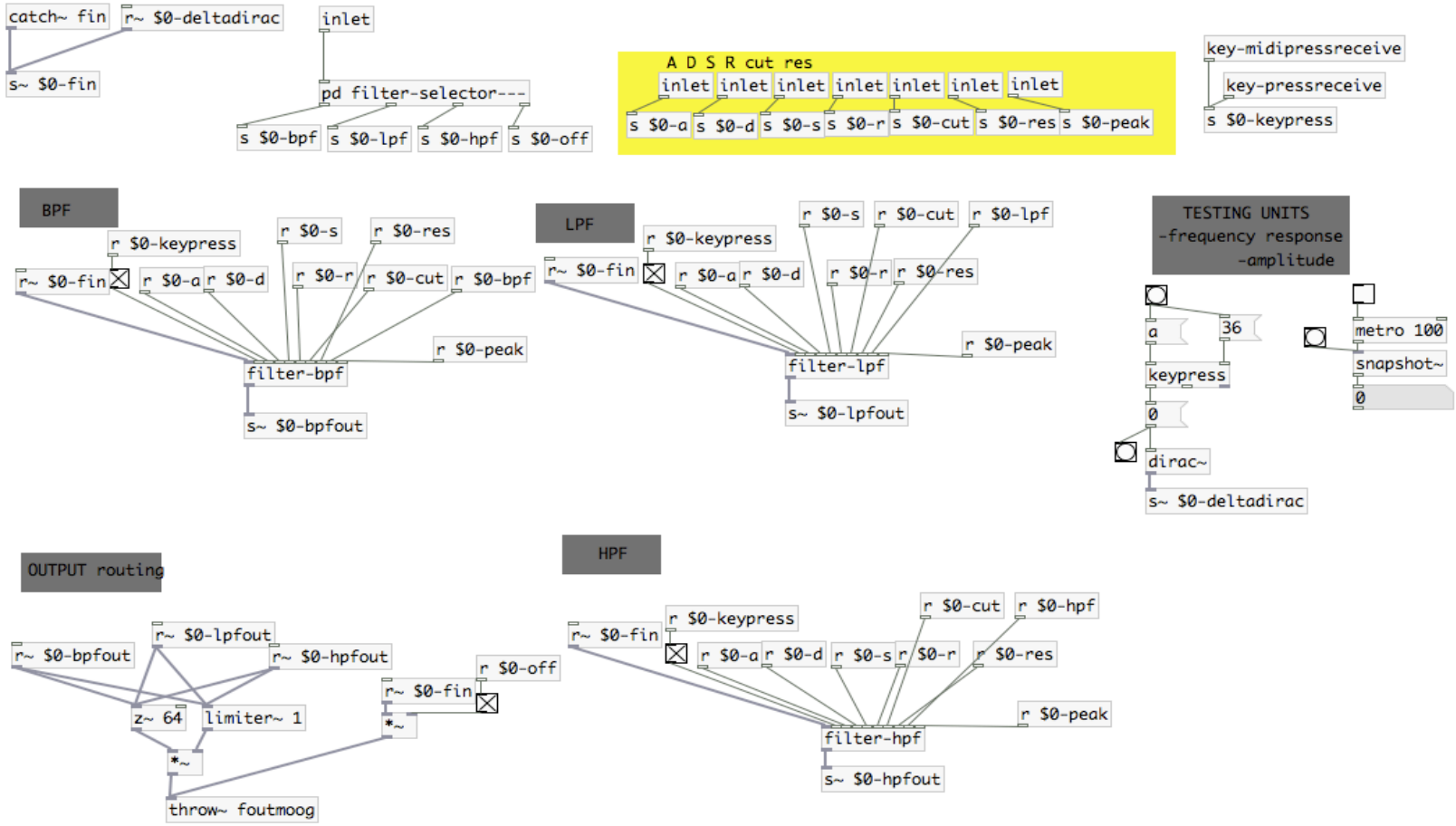


3. ADSR modulated Filter Bank abstraction

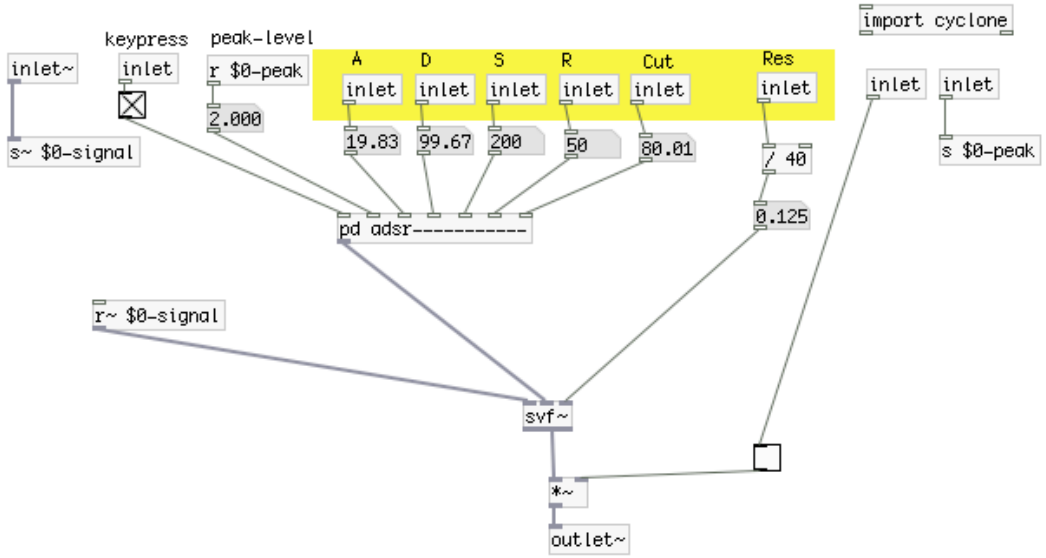
filter_interface.pd



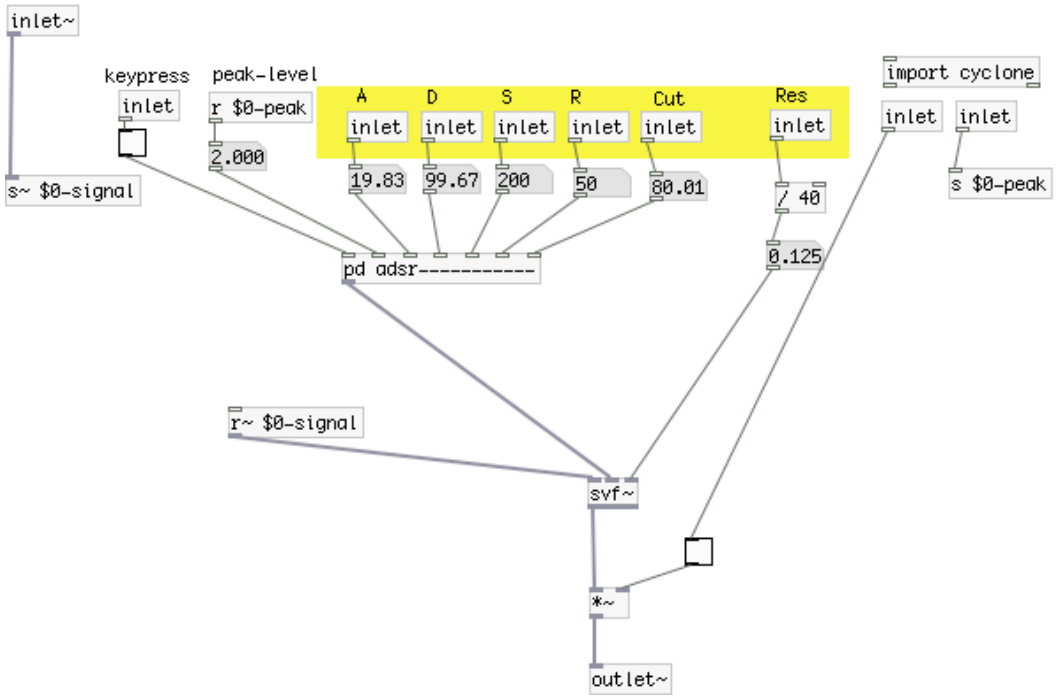
filter-bank.pd



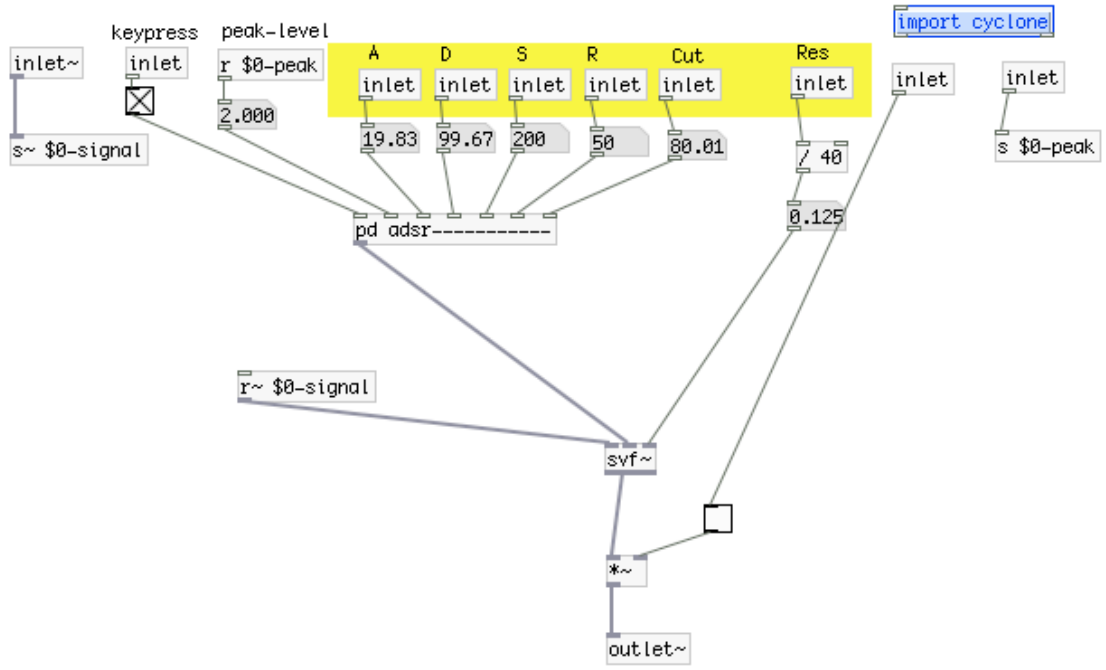
filter-bpf.pd



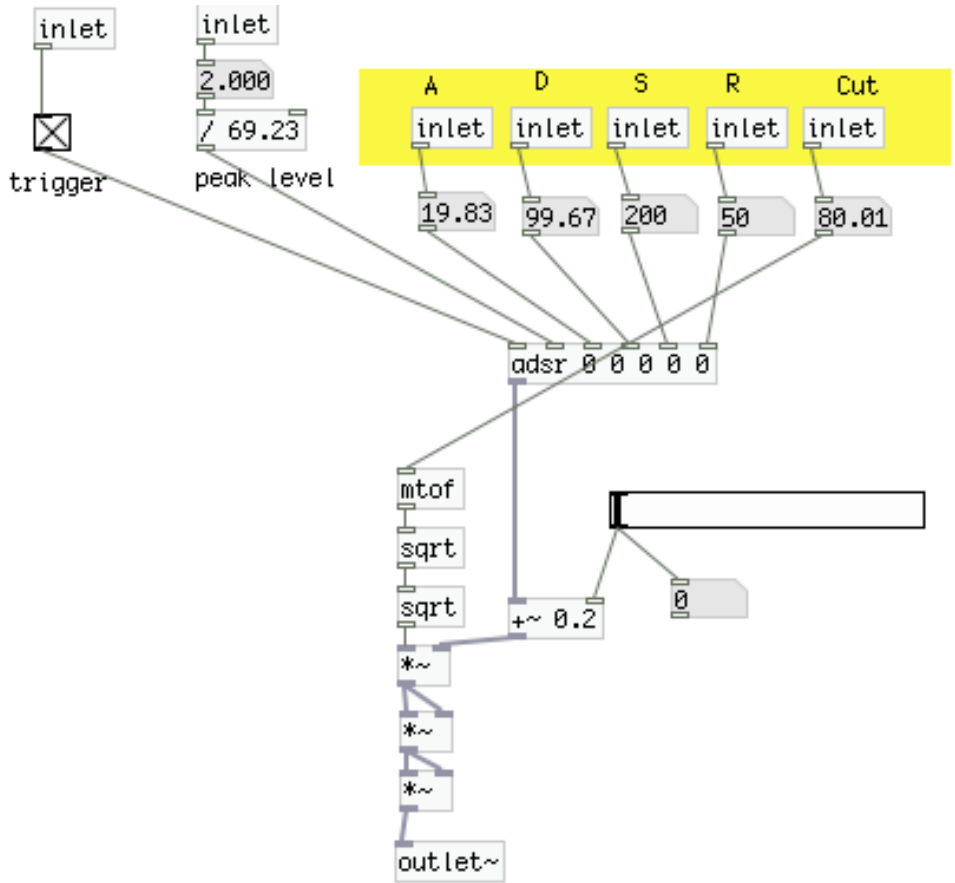
filter-lpf.pd



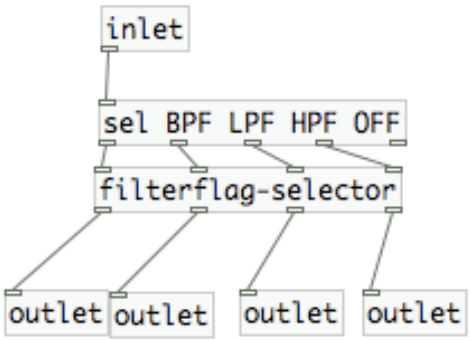
filter-hpf.pd



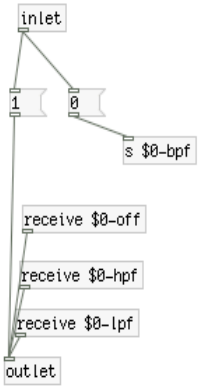
Subpatch: *pd adsr (in filter-hpf)*

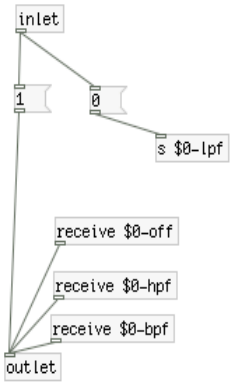


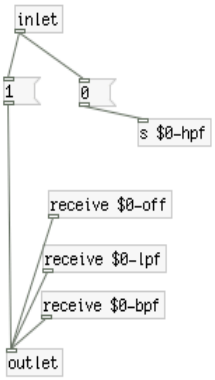
filter-selector.pd

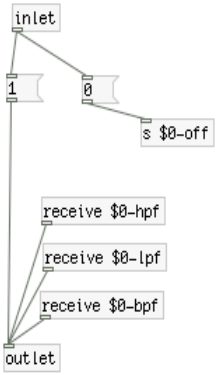


filterflag-selector.pd

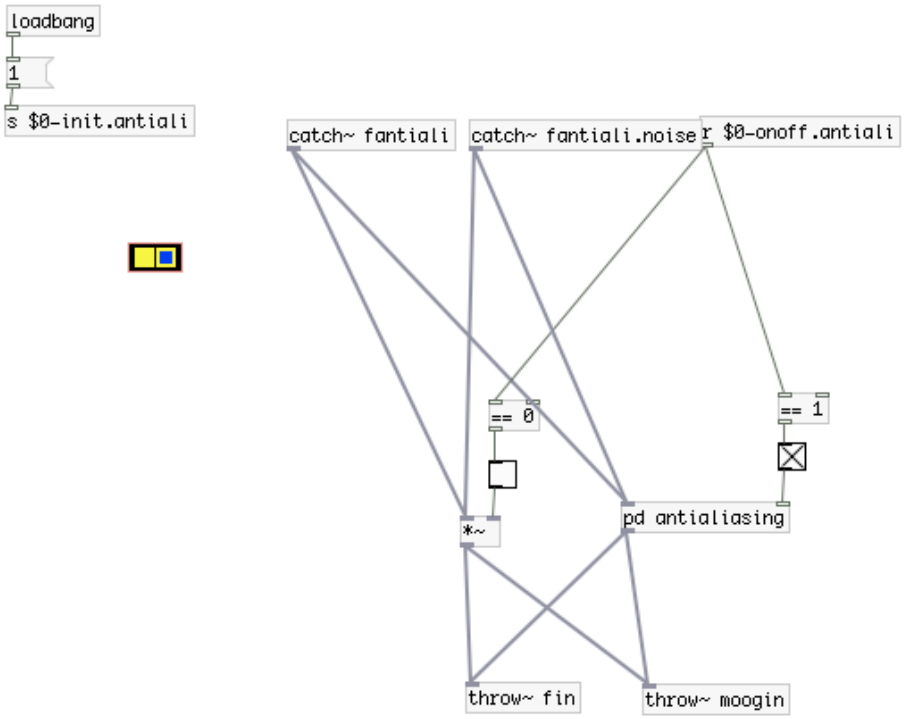




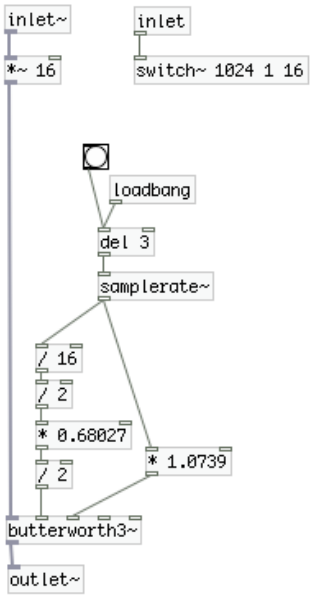




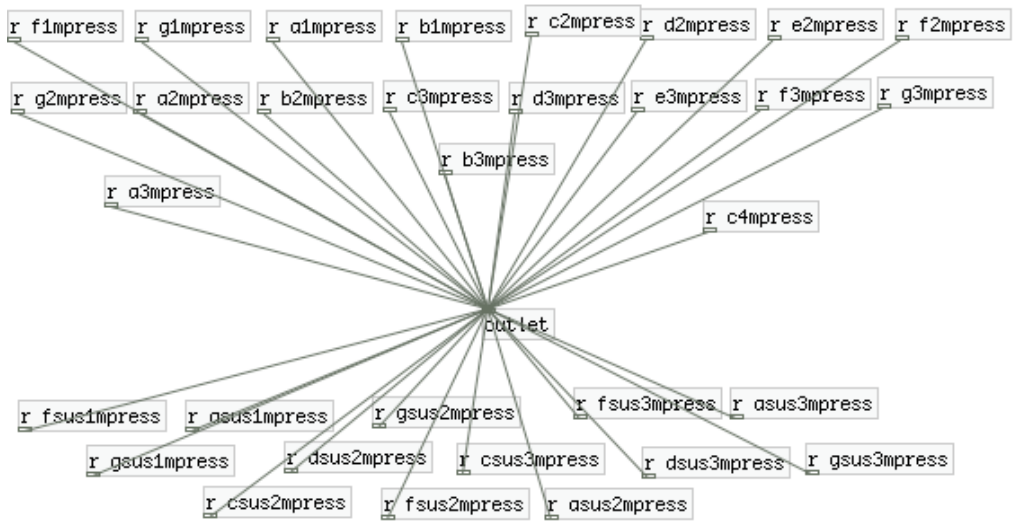
filter-antialiasing.pd



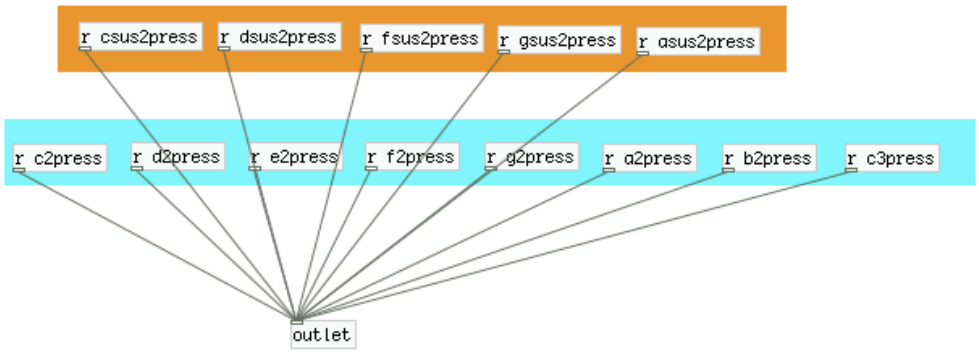
Subpatch oversampled: *pd antialiasing*



key-midipressreceive.pd

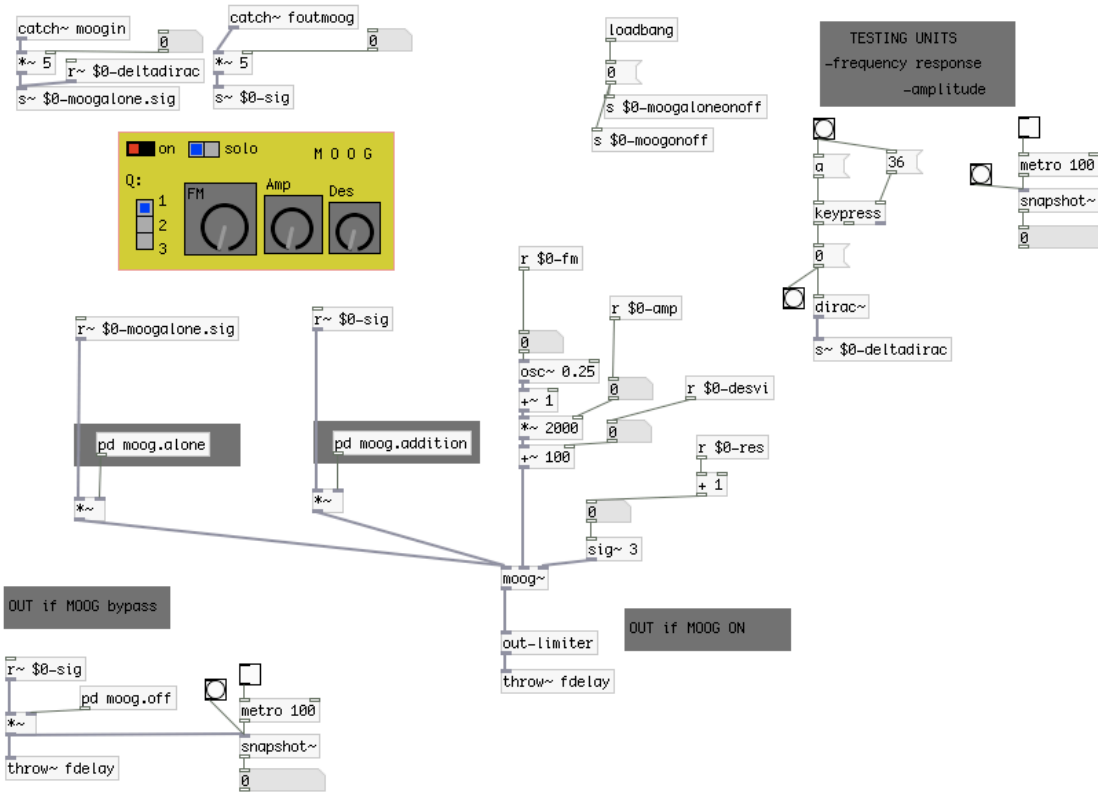


key-pressreceive.pd

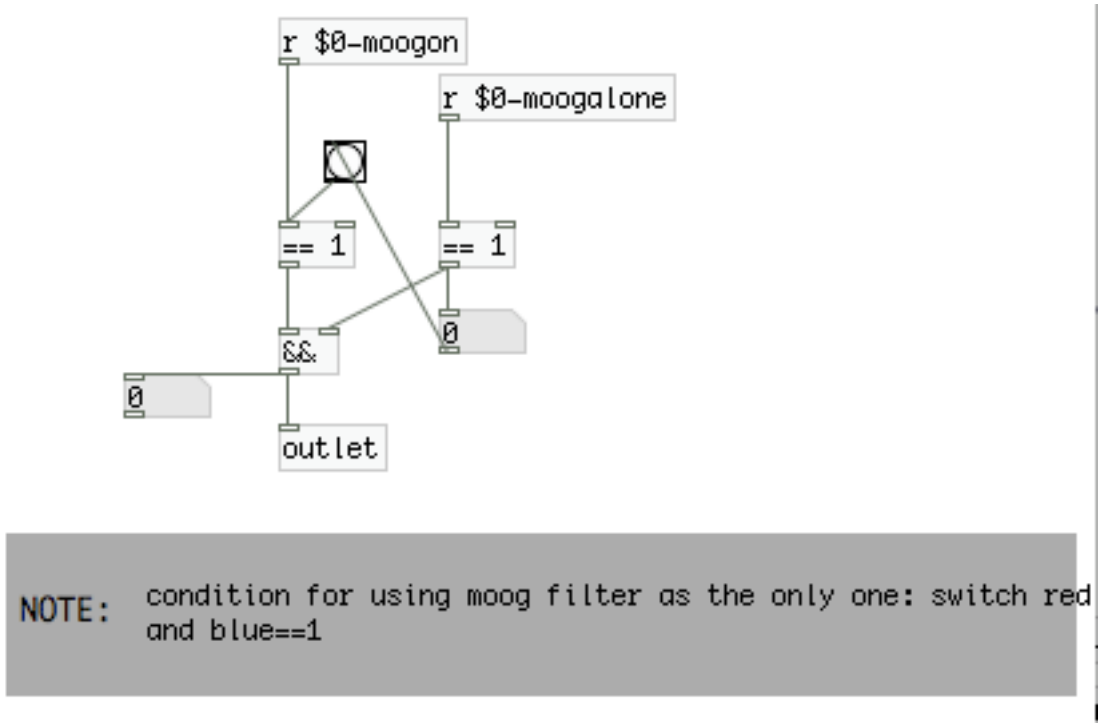


4. MOOG Filter abstraction

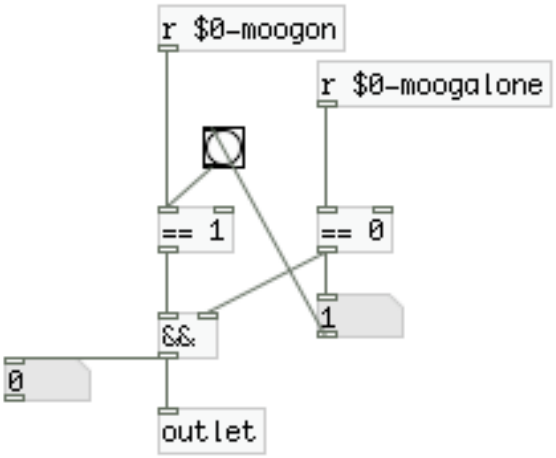
filter-moog.pd



Subpatch: *pd.moog.alone*



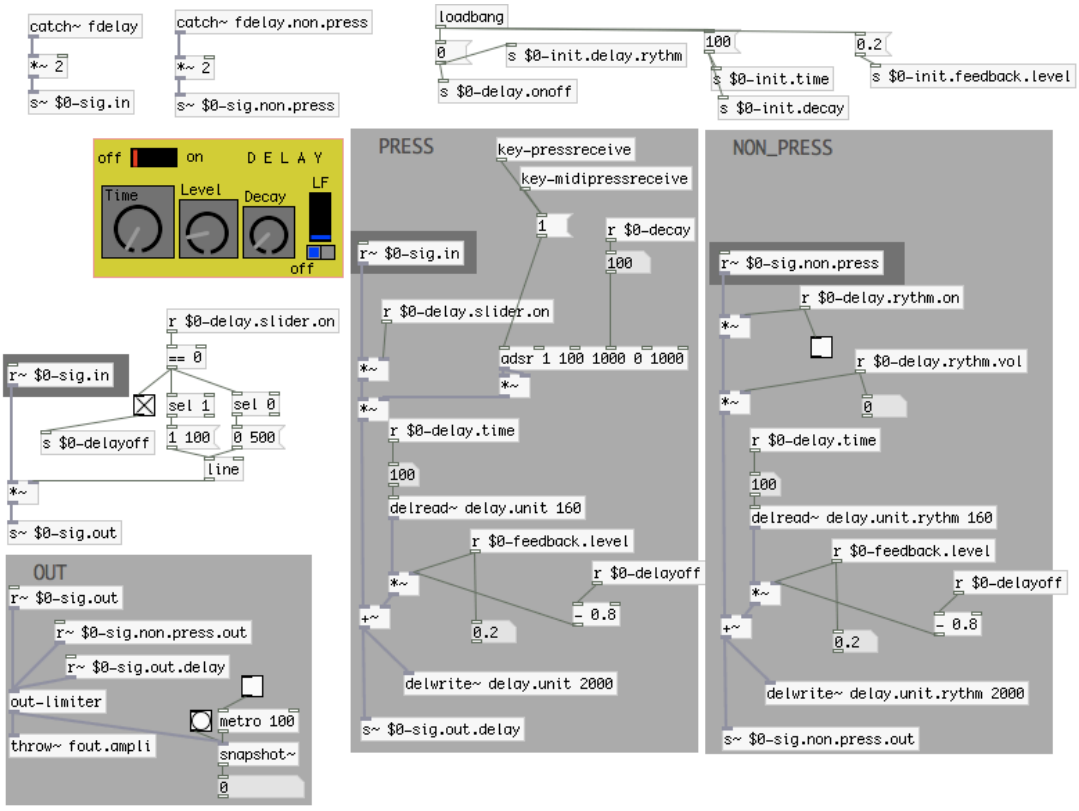
Subpatch: *pd moog.addition*



NOTE: condition for using moog filter added to the filterbank: switch red==1 and blue==0

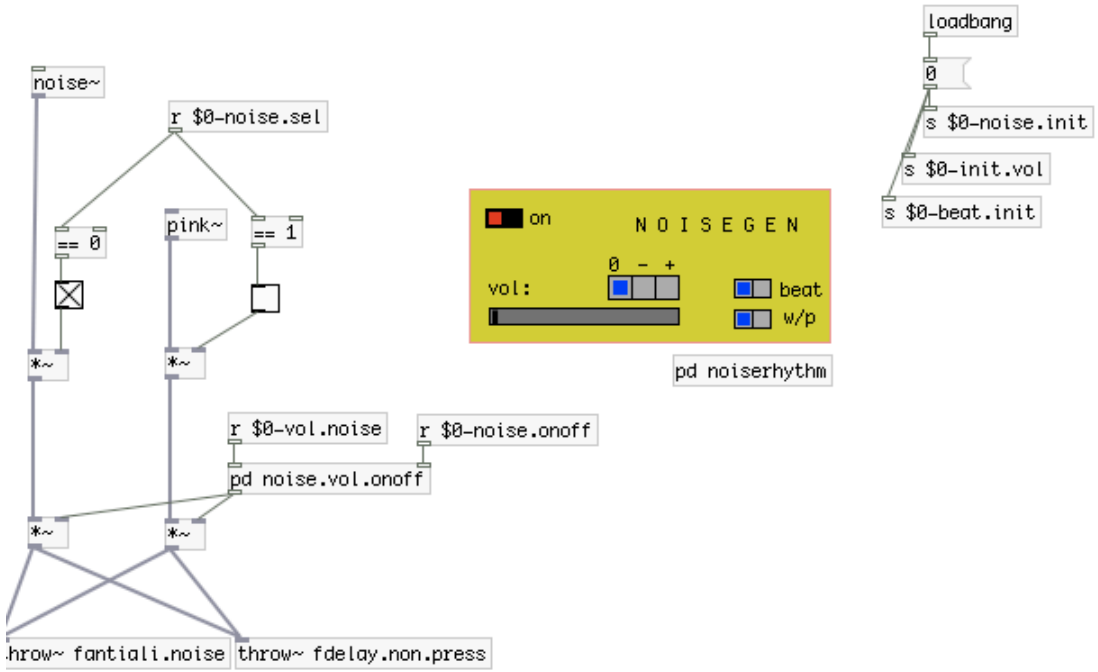
5. DELAY Unit abstraction

delay-unit.pd

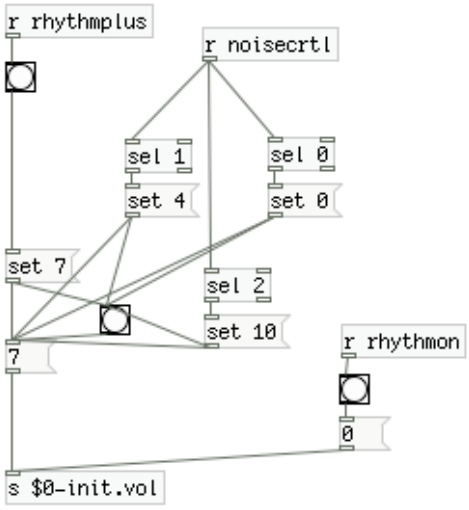


6. NOISE Generator Unit abstraction

noise.pd

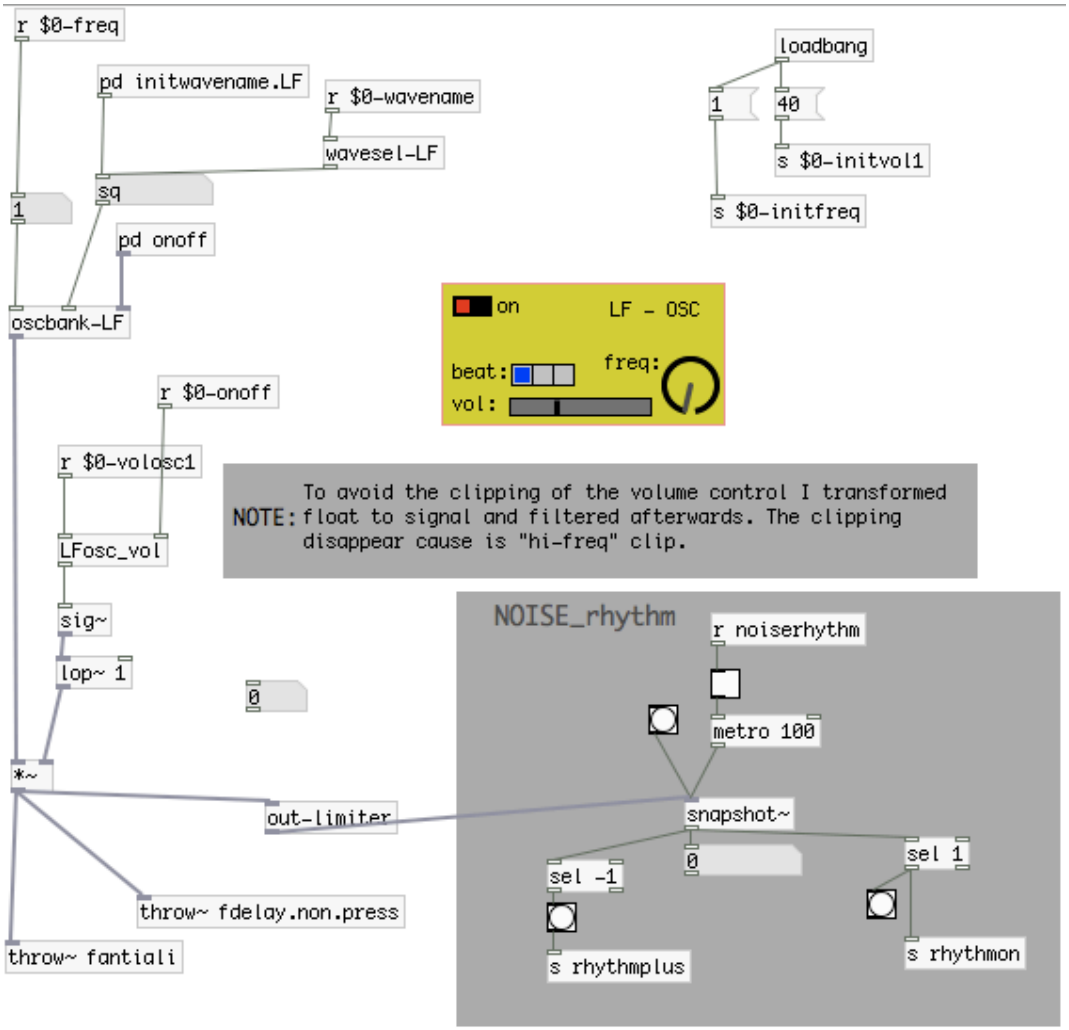


Subpatch: *pd noiserhythm*

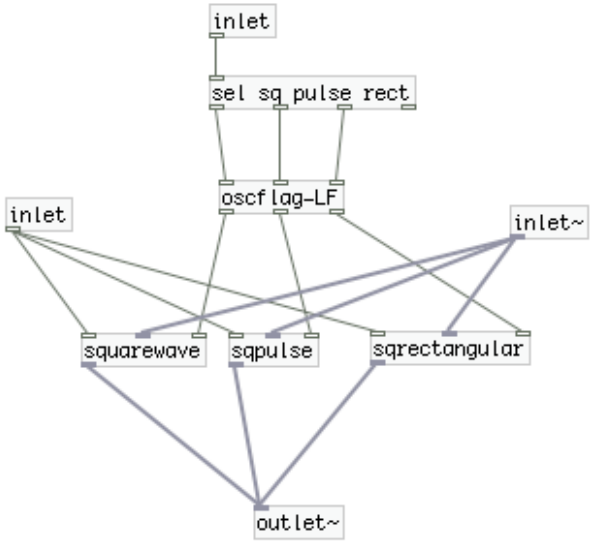


7. Low Frequency Oscillator Unit abstraction

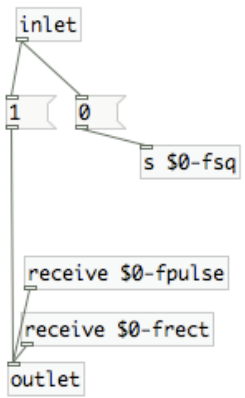
osc-LF.pd

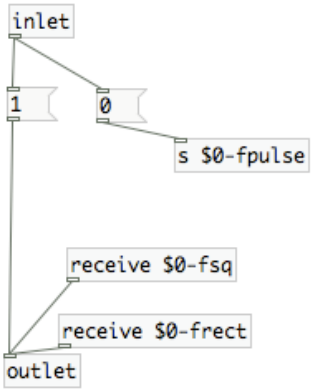


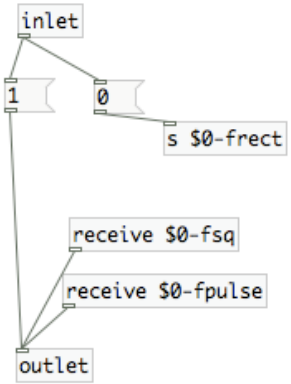
oscbank-LF.pd



oscflag-LF.pd

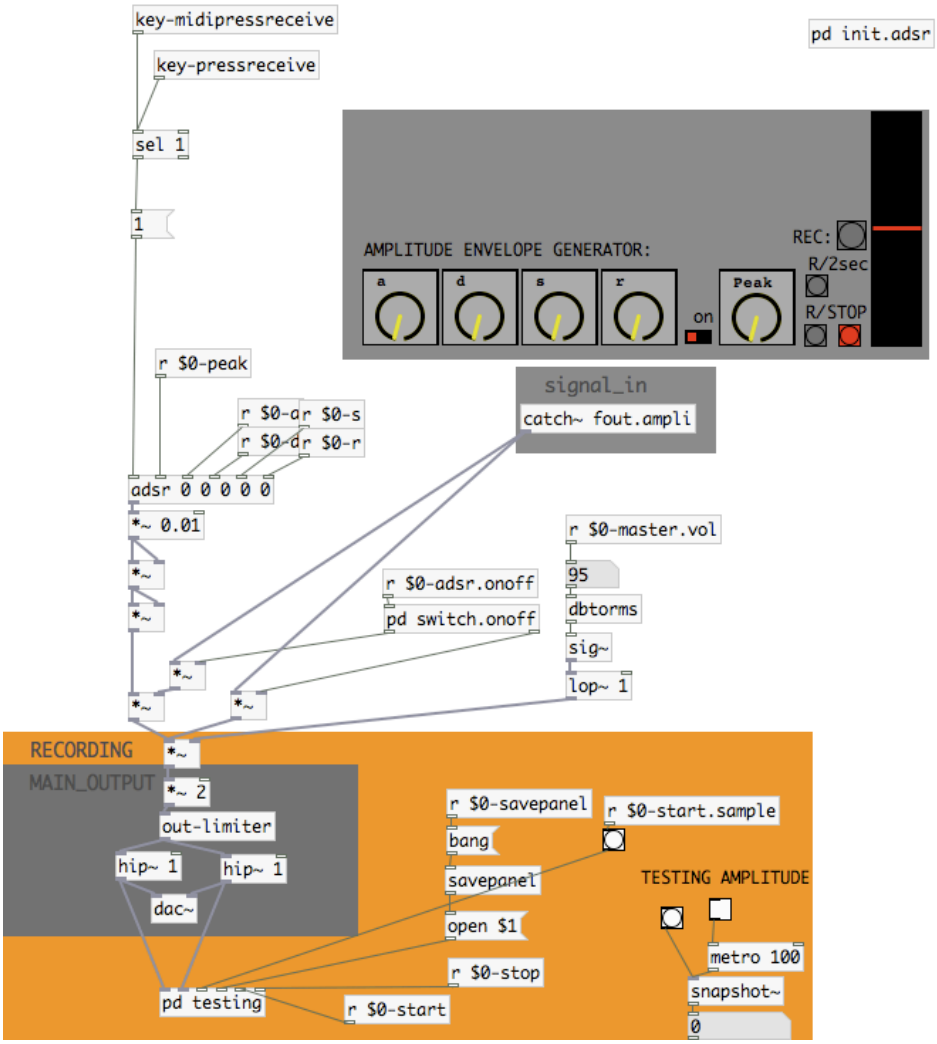




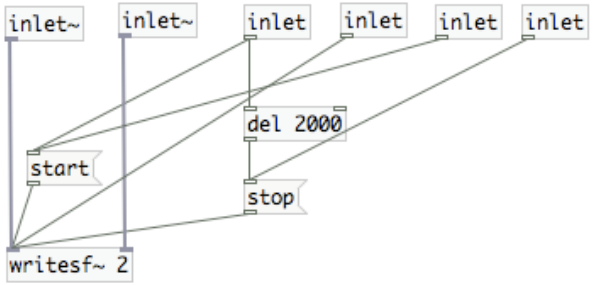


8. Main Output and ADSR Envelope Generator Amplifier abstraction

mainout-envelope.pd

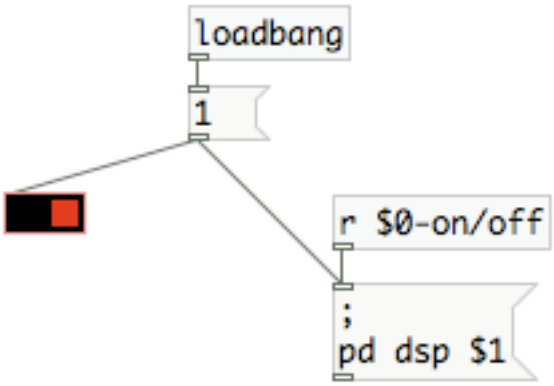


Subpatch: *pd testing*



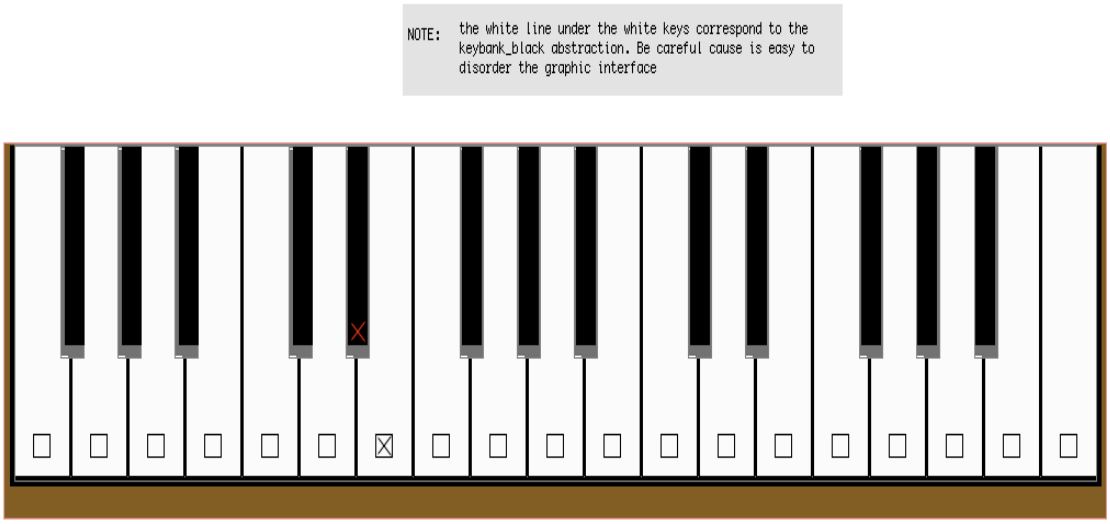
9. Main Switch abstraction

switch-on.pd

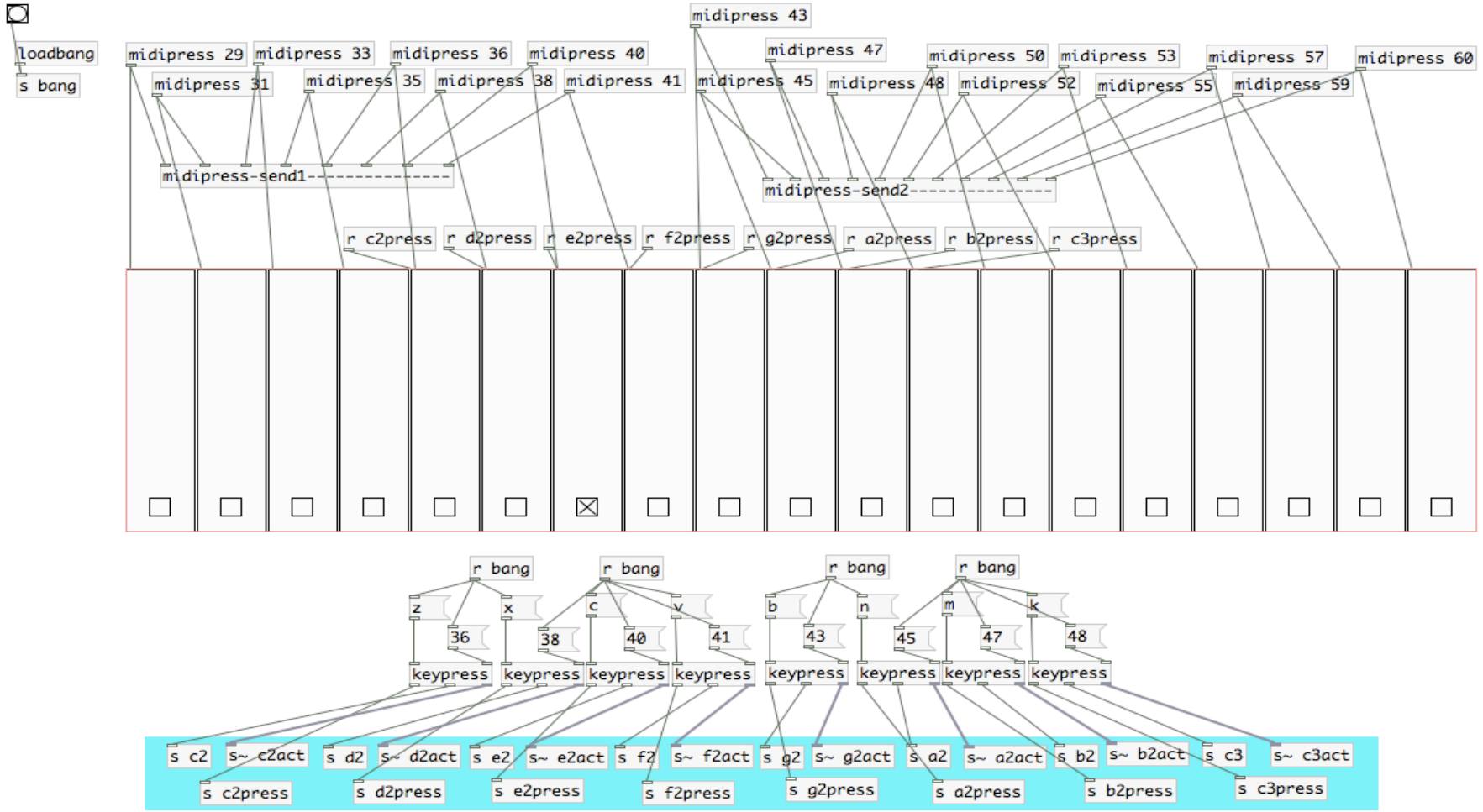


10. Keyboard abstraction

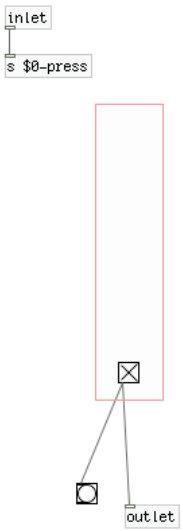
keyboard.pd



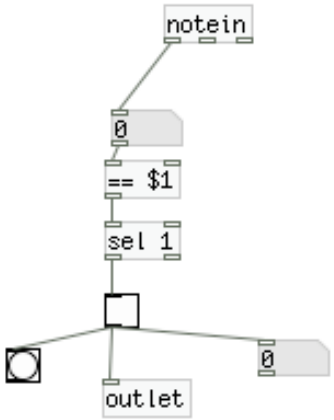
keybank_white.pd



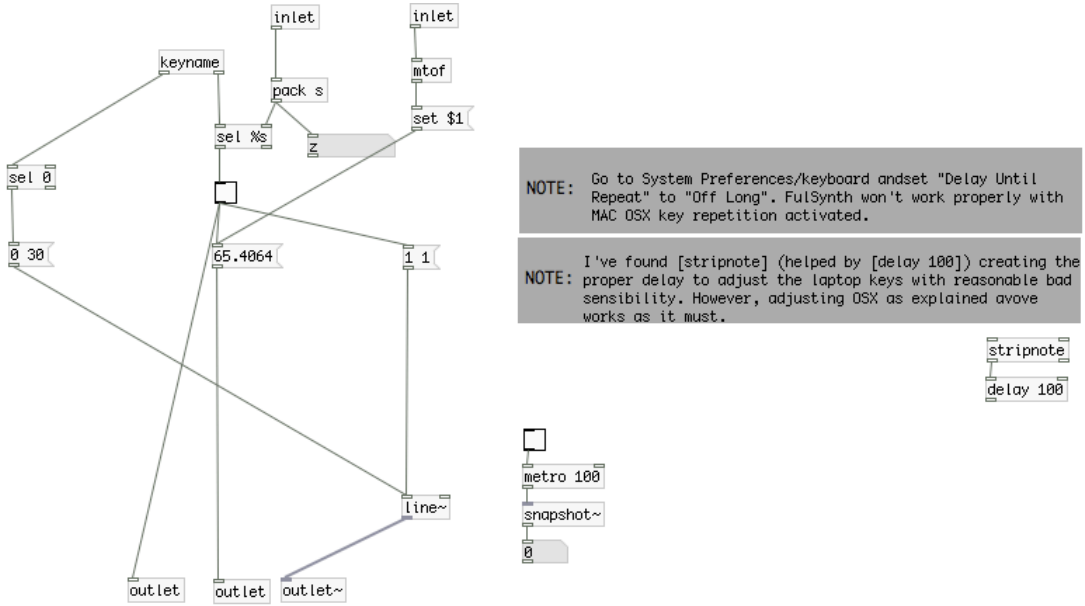
key_white.pd



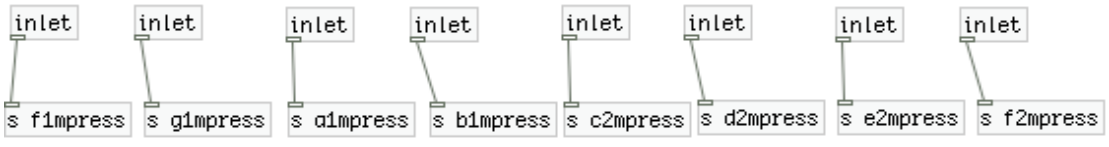
midipress.pd



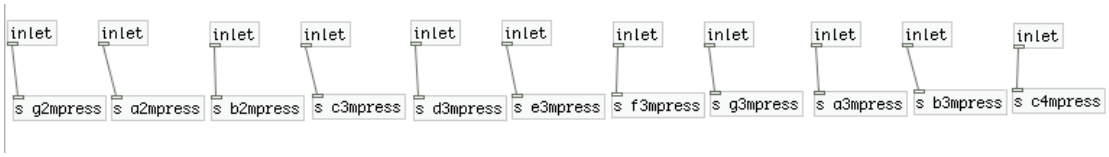
keypress.pd



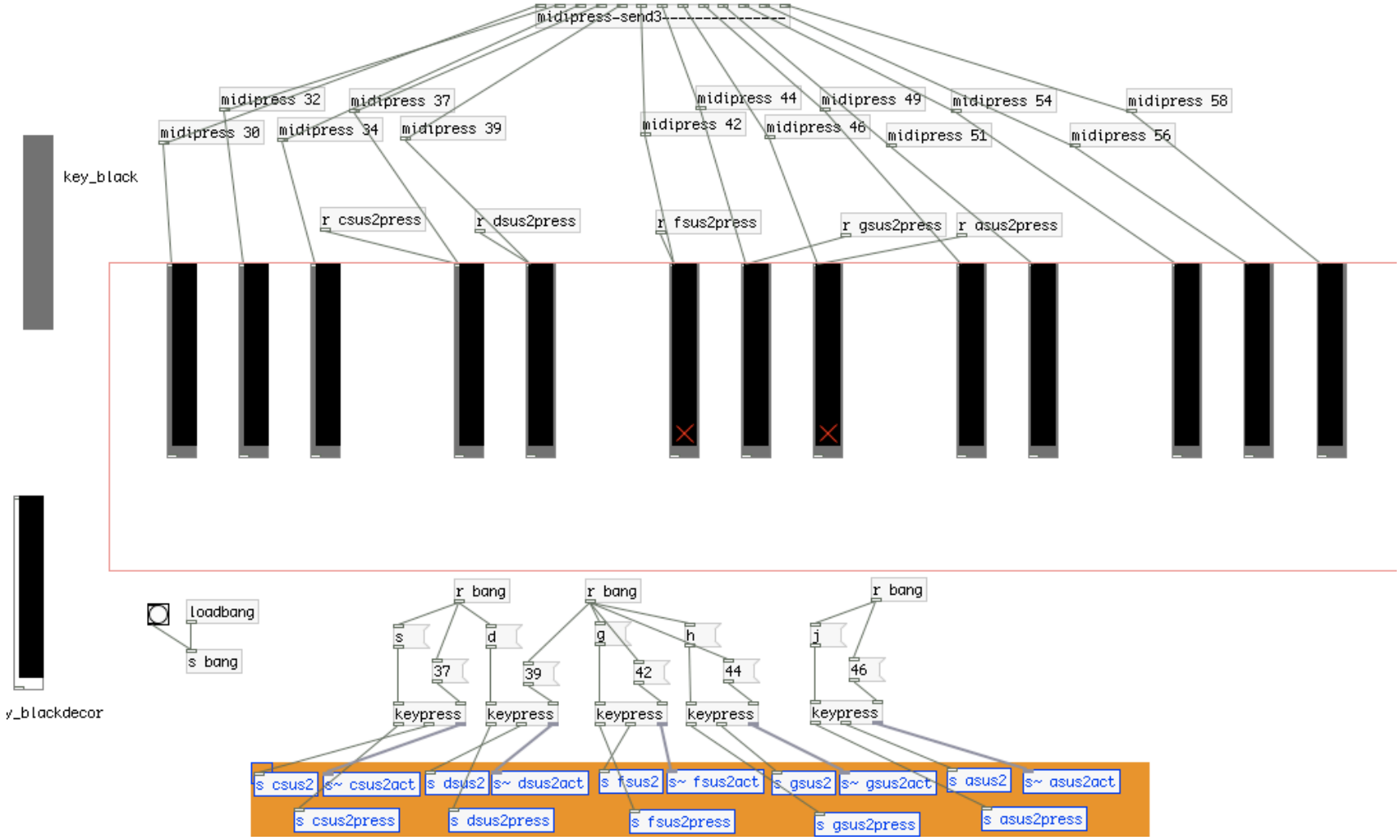
midipress-send1.pd



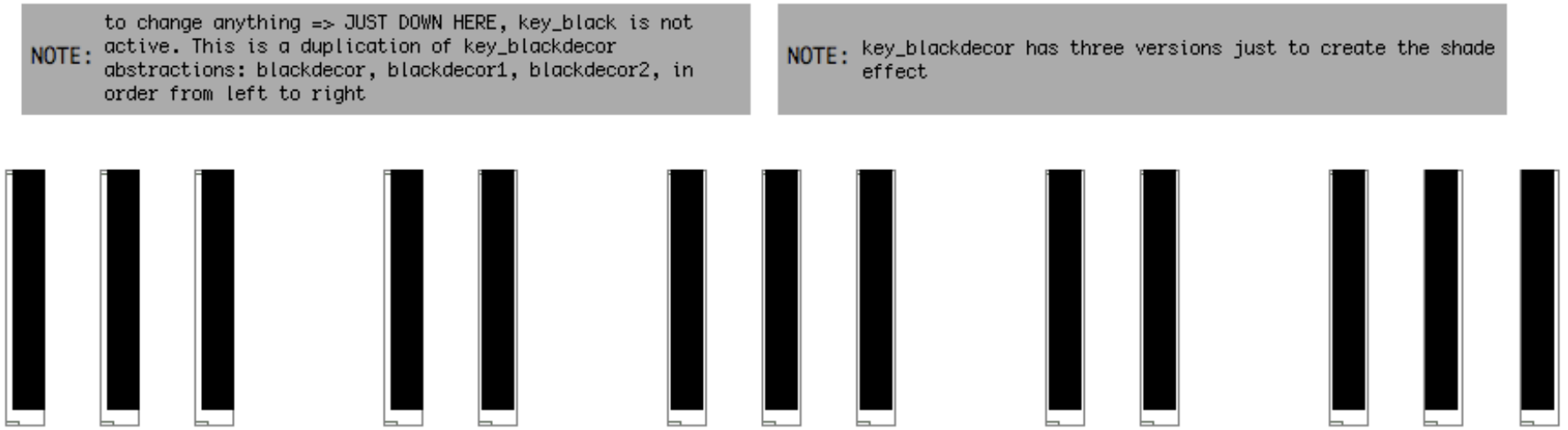
midipress-send2.pd



keybank_black.pd (part1)



keybank_black (part2)



key_blackdecor1.pd